ALGORYTMY WYZNACZANIA METRYK I MIAR CENTRALNOŚCI SIECI

Karol Działowski
Zachodniopomorski Uniwersytet Technologiczny

27 października 2020

1 Zadanie wprowadzające

Celem zadania drugiego było zapoznanie się z przykładami analizy miar sieciowych i ich interpretacja na mini grafach. W poniższych przykładach przeprowadzono:

1. Wyznaczanie średnicy sieci nieskierowanych, ważonych, skierowanych.
2. Nadawanie etykiet i identyfikatorów węzłom.
3. Wyznaczanie ścieżek dla grafów nieskierowanych, skierowanych i ważonych.
4. Wyznaczanie closeness centrality.
5. Wyznaczanie betweeness centrality.
6. Wyznaczanie eigenvectors i transitivity (clustering coefficient).

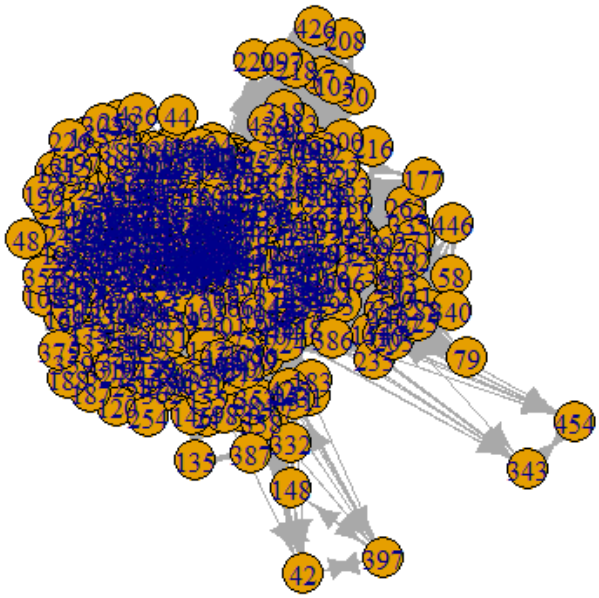
Wyniki zadania wprowadzającego zamieściłem w apendiksie A.

2 Analiza połączeń lotniczych

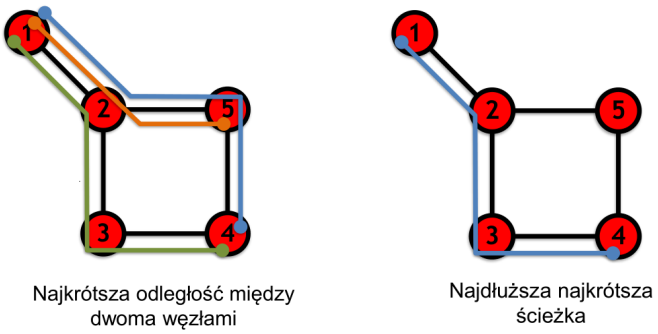
W tej części przeprowadzono analizę połączeń lotniczych. Skorzystano z sieci przedstawiającej połączenia lotnicze pomiędzy miastami w Stanach Zjednoczonych i Kanadzie.[1] Na rysunku 1 przedstawiono wizualizację sieci.

2.1 Średnica sieci i średnia najkrótsza ścieżka

Średnica grafu to najdłuższa najkrótsza ścieżka z kolei najkrótsza ścieżka to najkrótsza odległość pomiędzy dwoma węzłami. Przestawiono to na poniższym rysunku 2.



Rysunek 1: Wizualizacja sieci reachability.



(a) Najkrótsza ścieżka.

(b) Średnica grafu.

Rysunek 2: Wyjaśnienie najkrótszej ścieżki i średnicy grafu. Źródło: Jarosław Jankowski Algorytmy wyznaczania metryk i miar centralności sieci.

```
> # Srednia najkrotsza sciezka dla grafu nieskierowanego
> average.path.length(graph, directed = FALSE)
[1] 1.635705
> # Srednica sieci w grafie skierowanym
> diameter(graph, directed=T, weight=NA)
[1] 3
```

```
> get.diameter(graph, directed = TRUE)
+ 4/456 vertices, named, from 7640fd4:
[1] 39  27  57  343
> # Srednia najkrotsza sciezka dla grafu skierowanego
> average.path.length(graph, directed = TRUE)
[1] 1.655143
```

Średnia najkrótsza ścieżka dla grafu skierowanego wynosi 1,63. To oznacza, że średnia liczba przesiadek z dowolnego lotniska do drugiego dowolnego lotniska w tej sieci wynosi prawie 2.

Średnica sieci wynosi 3. Jest to to najdłuższa najkrótsza ścieżka. To znaczy, że z dowolnego węzła można dojść do innego maksymalnie przechodząc przez 3 połączenia.

2.2 Rozkłady

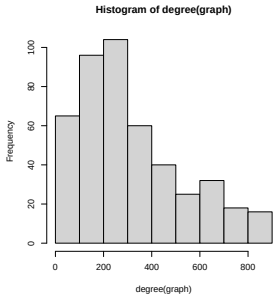
Stopień wierzchołka to liczba jego krawędzi.

Closeness to miara bliskości – średnia długość najkrótszej ścieżki między węzłem, a wszystkimi innymi węzłami. Bardziej centralne węzły mają bliżej do wszystkich innych węzłów w sieci.

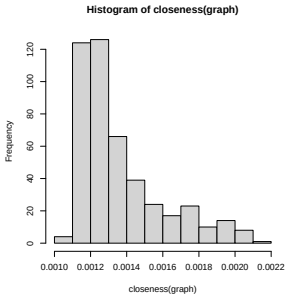
$$C(x) = \frac{1}{\sum_y d(y,x)} \tag{1}$$

Betweenness centrality to miara pośrednictwa, która określa ile razy węzeł stanowi pomost jako element najkrótszej ścieżki pomiędzy innymi węzłami sieci.

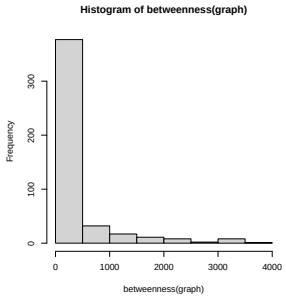
$$C_B(v) = \sum_{s \neg v \neg t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{2}$$



(a) Rozkład stopni wierzchołków.



(b) Rozkład closeness.



(c) Rozkład betweenness.

Rysunek 3: Rozkłady wierzchołków

Rozkład stopni wierzchołków pokazuje nam, że najwięcej liczny przedział jest dla liczby od 200 do 300 wierzchołków. Liczba wierzchołków w kolejnych grupach maleje.

Z rozkładu bliskości wynika, że wszystkie węzły są w bliskości z pozostałymi węzłami. Natomiast z analizy histogramu betweenness wynika, że większość połączeń jest nadmiarowych i węzły nie stanowią pomostu jako elementu najkrótszej ścieżki pomiędzy innymi węzłami sieci.

2.3 Wyznaczanie kluczowych węzłów

```
> dg = sort(degree(graph), decreasing = TRUE)[1:5]
> shortest.paths(graph, names(dg), names(dg), mode="out")
    246 368 230 74 94
246   0   1   1  1  1
368   1   0   1  1  1
```

```
230   1   1   0  1  1
74    1   1   1  0  1
94    1   1   1  1  0
>
> cl = sort(closeness(graph), decreasing = TRUE)[1:5]
> shortest.paths(graph, names(cl), names(cl), mode="out")
    246 230 74 368 100
246   0   1  1   1   1
230   1   0  1   1   1
74    1   1  0   1   1
368   1   1  1   0   1
100   1   1  1   1   0
>
> bt = sort(betweenness(graph), decreasing = TRUE)[1:5]
> shortest.paths(graph, names(bt), names(bt), mode="out")
    246 100 294 416 368
246   0   1   1   1   1
100   1   0   1   1   1
294   1   1   0   1   1
416   1   1   1   0   1
368   1   1   1   1   0
>
> intersect(intersect(names(dg),names(cl)),names(bt))
[1] "246" "368"
```

Powyżej wyznaczono węzły o największych wartościach stopnia, closeness i betweenness. Dla każdej pary węzłów wyznaczono najkrótsze ścieżki pomiędzy danymi węzłami. W każdym przypadku najkrótsza ścieżka wynosiła 1, czyli były to połączenia bezpośrednie. Wspólnymi węzłami dla wszystkich tych rankingów były węzły 246 i 368.

Literatura

[1] Brendan J. Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

A Zadanie wprowadzające - przebieg

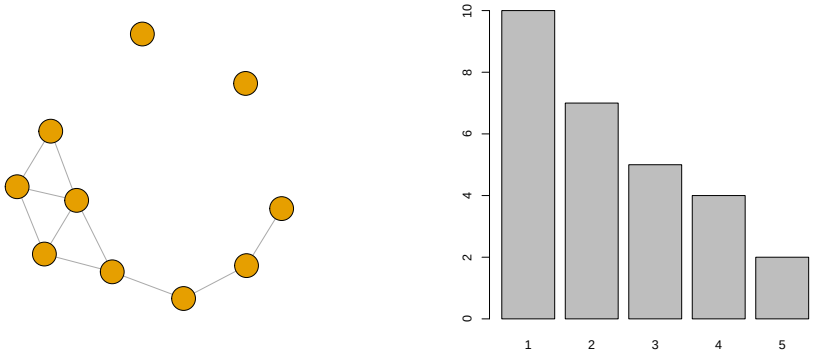
A.1 Średnica i najkrótsze ścieżki

Średnica grafu to najdłuższa najkrótsza ścieżka z kolei najkrótsza ścieżka to najkrótsza odległość pomiędzy dwoma węzłami. Przestawiono to na rysunku 2.

```
> graph <- sample_gnm(n=10, m=10)
> plot(graph, vertex.label=NA, vertex.size=18, vertex.label.cex=1.3)
>
> # Srednica sieci
> d = diameter(graph, directed=F, weights=NA)
>
> # Najkrotsze sciezki
> shortest.paths(graph)
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    0    2    4    3    1    3  Inf    1    4   Inf
 [2,]    2    0    2    1    1    1  Inf    3    2   Inf
 [3,]    4    2    0    2    3    1  Inf    5    1   Inf
...
 [9,]    4    2    1    1    3    1  Inf    5    0   Inf
[10,]  Inf  Inf  Inf  Inf  Inf  Inf  Inf  Inf  Inf     0
>
> # Najkrotsza sciezka pomiedzy wezlami 1 i 10
> shortest.paths(graph, 1, 10)
     [,1]
[1,]  Inf
> average.path.length(graph)
```

```
[1] 2.321429
> path.length.hist(graph)
$res
[1] 10  7  5  4  2

$unconnected
[1] 17
> # res - histogram
> # unconnected - liczba par, dla ktorych pierwszy wezel nie jest dostepny z drugiego
>
> tab <- as.table(path.length.hist(graph)$res)
> names(tab) <- 1:length(tab)
> barplot(tab)
```



(a) Wygenerowany graf.

(b) Rozkład najkrótszych ścieżek.

A.2 Ścieżki w grafach skierowanych

```
> ## S4 Sciezki w grafach skierowanych
> graph <- sample_gnm(n=10, m=10, directed = TRUE)
> plot(graph, vertex.size=18, vertex.label.cex=1.3)
> # Srednica sieci
> diameter(graph, directed=T, weight=NA)
[1] 3
> get.diameter(graph, directed = TRUE)
+ 4/10 vertices, from addf7bc:
[1]  5 10  3  1
>
> # Najkrotsze sciezki
> shortest.paths(graph, mode="out")
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    0  Inf  Inf  Inf  Inf  Inf  Inf  Inf  Inf   Inf
...
[10,]    2  Inf    1  Inf  Inf  Inf  Inf    2  Inf     0
> shortest.paths(graph, mode="in")
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    0  Inf    1  Inf    3  Inf  Inf    2  Inf     2
...
[10,]  Inf  Inf  Inf  Inf    1  Inf  Inf  Inf  Inf     0
> shortest.paths(graph, mode="all")
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]    0    4    1  Inf    3  Inf    5    2  Inf     2
...
[10,]    2    2    1  Inf    1  Inf    3    2  Inf     0
> shortest.paths(graph, 8, 10, mode="out") # zamiast FROM TO numery wewzlow
```

```
     [,1]
[1,]  Inf
```

A.3 Odczyt grafów skierowanych i wag krawędzi

```
> ## S5 Odczyt grafw skierowanych i wag krawdzi
> setwd("C:/Dev/complex_networks/lab_3/")
> graph <- read.graph("data/directed.txt", format="edgelist")
>
> # Graf skierowany - konwersja z data frame df
> df <- read.table("data/directed.txt", sep=" ", header = FALSE)
> df
   V1 V2
1   0  1
...
23  7  8
> graph <- graph.data.frame(df)
> shortest.paths(graph, mode="out")
  0 1 8 2 3 4 5 6 7
0 0 1 1 1 2 1 2 2 1
...
7 2 2 1 3 3 3 2 1 0
> shortest.paths(graph, mode="in")
  0 1 8 2 3 4 5 6 7
0 0 1 1 5 4 3 3 2 2
...
7 1 2 2 4 3 2 1 1 0
> shortest.paths(graph, mode="all")
  0 1 8 2 3 4 5 6 7
0 0 1 1 1 2 1 2 2 1
...
7 1 2 1 2 3 2 1 1 0
> shortest.paths(graph, 1, 7, mode="out") # zamiast FROM TO numery wewzlow
  5
0 2
```

A.4 Średnica z uwzględnieniem wag

```
> ## S6 Srednica z uwzglednieniem wag
> df <- read.table("data/weighted.txt", sep = " ", header = TRUE)
> graph <- graph.data.frame(df, directed = FALSE)
> plot(graph, edge.label = paste(df$weight1, df$weight2, sep=" "))
>
> edge.attributes(graph)
$weight1
 [1] 0.28 0.12 0.76 0.13 0.62 0.85 0.47 0.67 0.60 0.34 0.75 1.00 0.86 0.64 0.69
[16] 0.06 0.99 0.48 0.48

$weight2
 [1] 0.55 0.90 0.27 0.99 0.84 0.86 0.56 0.86 0.02 0.48 0.83 0.47 0.79 0.01 0.42
[16] 0.70 0.26 0.76 0.76

> E(graph)$weight1 <- df$weight1
> E(graph)$weight2 <- df$weight2
>
> graph <- set.edge.attribute(graph, "weight1", value=df$weight1)
> graph <- set.edge.attribute(graph, "weight2", value=df$weight2)
>
> # Wazona srednica
> diameter(graph, directed = TRUE, unconnected = TRUE, weights = df$weight1)
[1] 1.37
> get_diameter(graph, weights = df$weight1)
```

```
+ 4/9 vertices, named, from fad4abd:
[1] 4 9 7 8
> diameter(graph, directed = TRUE, unconnected = TRUE, weights = df$weight2)
[1] 1.27
> get_diameter(graph, weights = df$weight2)
+ 4/9 vertices, named, from fad4abd:
[1] 2 3 5 7
```

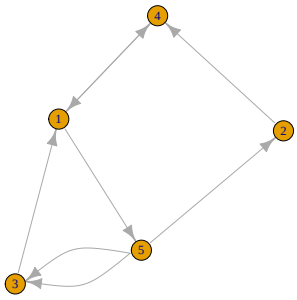
A.5 Grafy nazwane i nienazwane i odwołania do węzłów.

```
> ## S7 Grafy nazwane i nienazwane i odwolania do wezlow
> # Identyfikatory wezlow graf nienazwany
> graph <- sample_gnm(n=10, m=10)
> V(graph)
+ 10/10 vertices, from 25a6ae1:
 [1]  1  2  3  4  5  6  7  8  9 10
> V(graph)$name
NULL
> plot(graph)
> is_named(graph)
[1] FALSE
>
> # Identyfikatory wezlow graf nazwany
> graph <- sample_gnm(n=10, m=10)
> set_vertex_attr("name", value = letter[1:10])
Bd w poleceniu 'i_set_vertex_attr(graph = graph, name = name, index = index, ':
  Not a graph object
> V(graph)
+ 10/10 vertices, from 25b2bbf:
 [1]  1  2  3  4  5  6  7  8  9 10
> V(graph)$name
NULL
> plot(graph, vertex.label=V(graph)$name)
> as.numeric(V(graph))
 [1]  1  2  3  4  5  6  7  8  9 10
>
> # Identyfikatory wczytywane z pliku
> df <- read.table("weighted.txt", sep = " ", header = TRUE)
> graph <- graph.data.frame(df, directed = FALSE)
> V(graph) # 9 1 2 3 4 czyli wezel z etykieta 9 ma id 1
+ 9/9 vertices, named, from 25e034a:
[1] 9 1 2 3 4 5 6 7 8
> V(graph)$name
[1] "9" "1" "2" "3" "4" "5" "6" "7" "8"
> as.numeric(V(graph))
[1] 1 2 3 4 5 6 7 8 9
> plot(graph, vertex.label=V(graph)$name)
> plot(graph, vertex.label=as.numeric(V(graph)))
> neighbors(graph, 1) # sasiedzi 9
+ 5/9 vertices, named, from 25e034a:
[1] 1 2 4 7 8
> neighbors(graph, "9")
+ 5/9 vertices, named, from 25e034a:
[1] 1 2 4 7 8
> neighbors(graph, 9)
+ 5/9 vertices, named, from 25e034a:
[1] 9 1 1 6 7
```

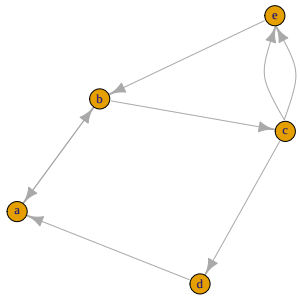
A.6 Alfanumeryczne nazwy węzłów w pliku.

```
> ## S8 Alfanumeryczne nazwy wezlow w pliku
> df <- read.table("data/weighted_abc.txt", sep = " ", header = TRUE)
```

```
> graph <- graph.data.frame(df, directed = TRUE)
> plot(graph, vertex.label=V(graph)$name, edge.label = paste(df$weight1, df$weight2, sep=" "))
> V(graph)
+ 5/5 vertices, named, from 88332c1:
[1] b d e a c
>
> shortest.paths(graph, 1, 2, weights=df$weight1, mode="out")
     d
b 1.09
> shortest.paths(graph, "a", "b", weights=df$weight1, mode="out")
     b
a 0.12
> plot(graph, vertex.label=V(graph)$name)
> plot(graph, vertex.label=as.numeric(V(graph)))
```



(a) Graf z indeksami.



(b) Graf z nazwami.

A.7 Ścieżki ważone i odwołania do węzłów.

```
> ## S9 Sciezki wazone i odwolania do wezlow
> df <- read.table("data/weighted.txt", sep = " ", header = TRUE)
> graph <- graph.data.frame(df, directed = TRUE)
> plot(graph, edge.label = paste(df$weight1, df$weight2, sep=" "))
> shortest.paths(graph, 1, weights=df$weight1, mode="out") # wynik wezla id=1 "9"
  9    1    2    3    4    5    6    7    8
9 0 0.28 0.12 0.72 0.76 1.47 0.75 0.13 0.61
> shortest.paths(graph, "9", weights=df$weight1, mode= "out") #wynik dla name="9"
  9    1    2    3    4    5    6    7    8
9 0 0.28 0.12 0.72 0.76 1.47 0.75 0.13 0.61
> shortest.paths(graph, 1, 7, weights=df$weight1, mode= "out") #wynik dla "9" i "6"
     6
9 0.75
> shortest.paths(graph, "1", "7", weights=df$weight1, mode= "out") #wynik dla "1" "7"
     7
1 0.53
> # in vs out weight1 np. koszt
> shortest.paths(graph, "2", "3", weights=df$weight1, mode= "in")
    3
2 Inf
> shortest.paths(graph, "2", "3", weights=df$weight1, mode= "out")
    3
2 0.6
> # in vs out weight2 np. czas
> shortest.paths(graph, "9", "5", weights=df$weight2, mode= "in")
    5
```

```
9 Inf
> shortest.paths(graph, "9", "5", weights=df$weight2, mode= "out")
     5
9 0.74
```

A.8 Closeness dla grafów skierowanych i ważonych

Closeness to miara bliskości – średnia długość najkrótszej ścieżki między węzłem, a wszystkimi innymi węzłami. Bardziej centralne węzły mają bliżej do wszystkich innych węzłów w sieci.

$$C(x) = \frac{1}{\sum_y d(y,x)} \tag{3}$$

```
> ## S10 Closeness dla grafow skierowanych i wazonych
> graph <- sample_gnm(n=10, m=20)
> plot(graph)
> closeness(graph)
 [1] 0.07142857 0.06666667 0.08333333 0.07692308 0.06250000 0.08333333
 [7] 0.07692308 0.05555556 0.06666667 0.06250000
> closeness(graph,vids=c("1","2"))
[1] 0.07142857 0.06666667
>
> df <- read.table("data/closeness2.txt", sep = " ", header = TRUE )
> graph <- graph.data.frame(df, directed = FALSE )
> plot(graph, vertex.label=V(graph)$name)
> closeness(graph)
         1          9          5          2          3          4          6
0.06666667 0.07142857 0.06666667 0.04545455 0.04545455 0.04545455 0.04545455
         7          8
0.04545455 0.04545455
> closeness(graph,vids=c("1", "9"))
         1          9
0.06666667 0.07142857
```

```
## S11 Closeness dla grafow skierowanych i wazonych
df <- read.table("data/closeness.txt", sep = " ", header = TRUE)
graph <- graph.data.frame(df, directed = FALSE )
plot(graph,edge.label = paste(df$weight1, df$weight2,sep=" "))
closeness(graph, weights = df$weight1)

df <- read.table("closeness.txt", sep = " ", header = TRUE )
graph <- graph.data.frame(df, directed = TRUE )
plot(graph,edge.label = paste(df$weight1, df$weight2,sep=" "))
closeness(graph)
```

A.9 Betweenness dla grafów skierowanych i ważonych.

Betweenness centrality to miara pośrednictwa, która określa ile razy węzeł stanowi pomost jako element najkrótszej ścieżki pomiędzy innymi węzłami sieci.

$$C_B(v) = \sum_{s \neg v \neg t \in V} \frac{\sigma_{st}(v)}{\sigma_{st}} \tag{4}$$

```
> ## S12 Betweenness dla grafow skierowanych i wazonych
> graph <- sample_gnm(n=10, m=20)
> plot(graph)
> betweenness(graph)
 [1] 3.9166667 2.7500000 1.7500000 1.6666667 4.5000000 0.0000000 2.5000000
 [8] 2.8333333 0.9166667 7.1666667
```

```
>
> df <- read.table("data/closeness2.txt", sep = " ", header = TRUE )
> graph <- graph.data.frame(df, directed = FALSE )
> plot(graph, vertex.label=V(graph)$name)
> betweenness(graph)
 1  9  5  2  3  4  6  7  8
18 16 18  0  0  0  0  0  0
> b <- betweenness(graph)
>
> betweenness(graph, v = V(graph), directed = TRUE, weights = NULL,
+             nobigint = TRUE, normalized = FALSE)
 1  9  5  2  3  4  6  7  8
18 16 18  0  0  0  0  0  0
```

A.10 Eigenvector i clustering coefficient

```
> ## S13 Eigenvector i clustering coefficient
> df <- read.table("data/directed.txt", sep = " ", header = TRUE )
> graph <- graph.data.frame(df, directed = FALSE )
> plot(graph, vertex.label=V(graph)$name)
> eigen_centrality(graph)
$vector
        1         0         8         2         3         4         5
0.7847817 0.8353442 0.9657926 0.2971644 0.3645355 0.5399779 0.6428215
        6         7
1.0000000 0.8441406

...

> transitivity(graph) # clustering coefficient
[1] 0.3529412
> page_rank(graph)
$vector
         1          0          8          2          3          4          5
0.11176860 0.13327811 0.12943265 0.07805349 0.10001356 0.09466312 0.09208051
         6          7
0.15066801 0.11004194

$value
[1] 1

$options
NULL
```