Techniki Analizy Sieci Społecznych (TASS)

Projekt 1. Analiza statystyczna grafu przy użyciu standardowych narzędzi

Autor: Mateusz Hryciów, 283365

1. Zadanie A

283365 mod 7 = 5, zatem zbiór danych to „Sieć interakcji pomiędzy delfinami”. Wszystkie zadania zostały wykonane przy użyciu programu Pajek w wersji 5.11.

1.1. Zbadaj, jaki jest rząd i rozmiar całej sieci, a następnie wyodrębnij największą składową spójną, zbadaj jej rząd i rozmiar

Po wczytaniu sieci do programu Pajek zbadano jaki jest rozmiar (liczba krawędzi) oraz rząd (liczba wierzchołków sieci).

```
Number of vertices (n): 62
----------------------------------------------------------
                                       Arcs         Edges
----------------------------------------------------------
Total number of lines                     0           159
----------------------------------------------------------
Number of loops                           0             0
Number of multiple lines                  0             0
----------------------------------------------------------
```

Rysunek 1. Rząd oraz rozmiar wczytanej sieci.

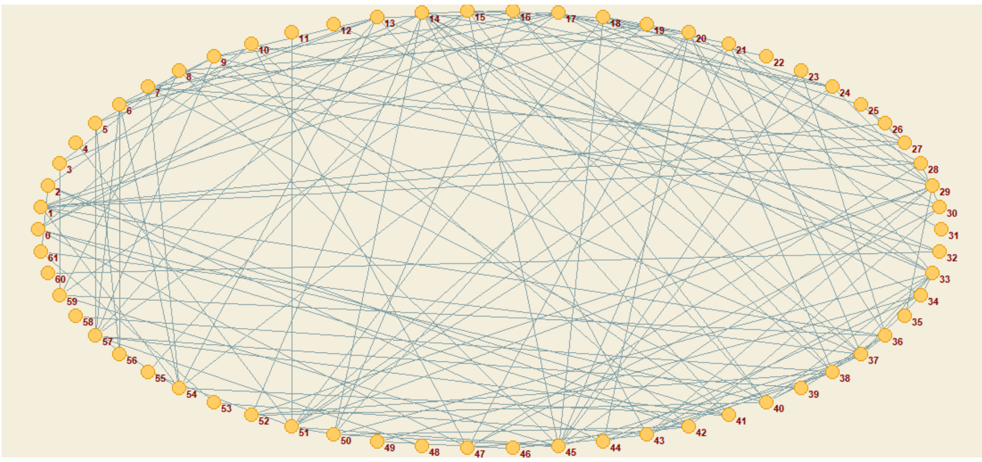
Otrzymane wyniki przedstawiono w tabeli 1.

Tabela 1. Parametry wczytanej sieci

Wczytana sieć	
Rząd	**Rozmiar**
62	159

Następnie wyodrębniono największą składową spójną. Okazało się, że rozmiar oraz rząd pozostały takie same. Oznacza to, że graf od początku był spójny, czyli jego każda para wierzchołków była połączona ścieżką.

1.2. Wykreśl największą składową spójną i skomentuj wynik

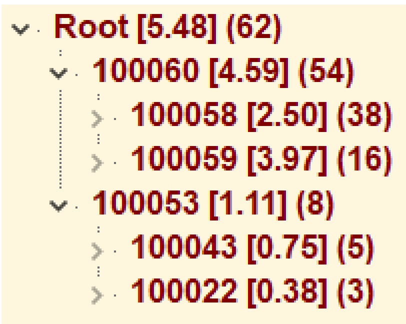


Rysunek 2. Składowa spójna.

Na podstawie rys. 2 można zauważyć, że stopnie wierzchołków są niewielkie, gdyż wychodzi z nich klika krawędzi. Jednakże są one zbliżone do siebie. Ponadto można stwierdzić, że graf nie jest pełny.

1.3. Przeprowadź grupowanie metodą Warda z metryką d1 (odległość dwóch węzłów to liczba sąsiadów połączonych tylko z jednym z nich)

Zgodnie z instrukcją, aby przeprowadzić grupowanie metodą Warda należy na początku stworzyć kompletny klaster, a następnie stworzyć drzewo hierarchii. Otrzymano:

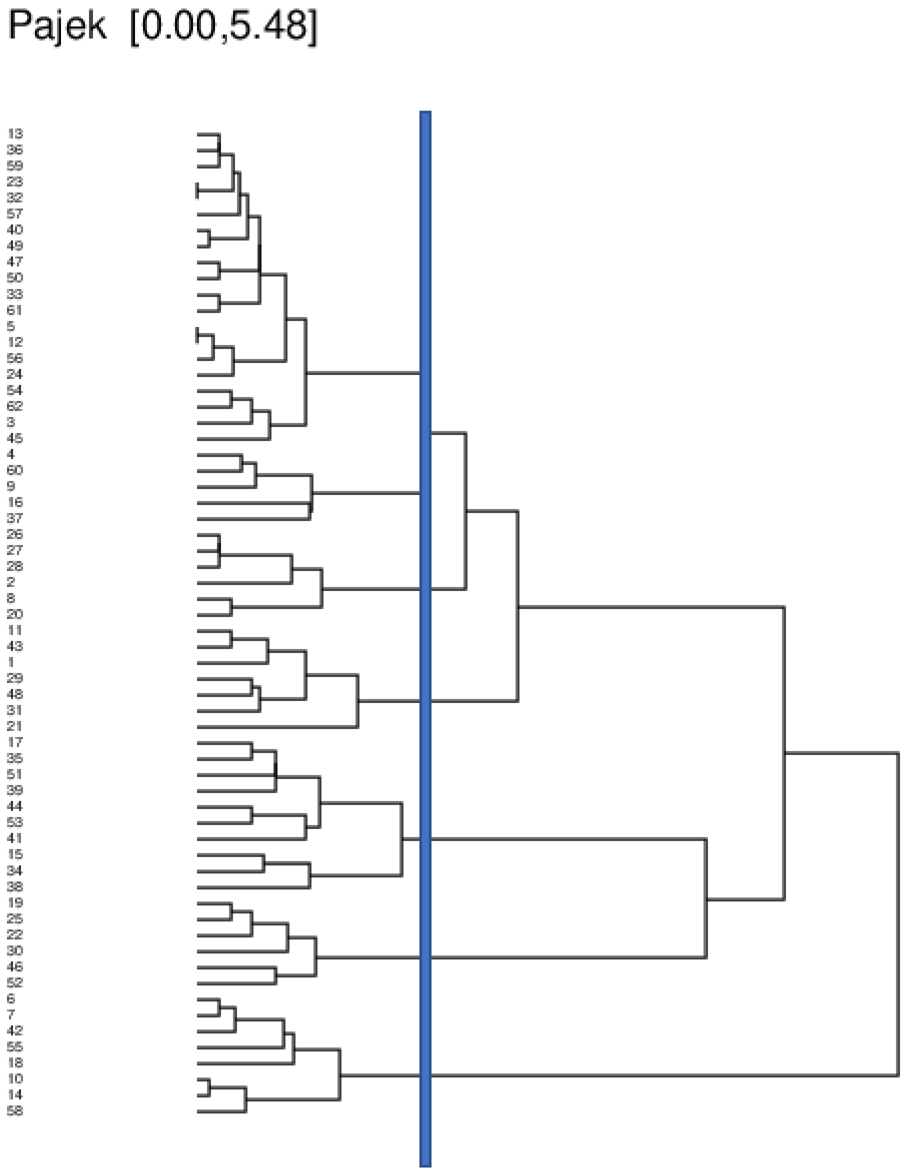


Rysunek 3. Wyniki grupowanie metodą Warda z metryką d1.

Zgodnie z teorią można zauważyć, że wraz z przesuwaniem się w głąb drzewa wartość metryki maleje. Im węzły są bardziej do siebie podobne tym mniejsza wartość. W przypadku idealnym wynosi ona 0. W badanym przypadku wartości metryki są duże, co może oznaczać, że różne wierzchołki sieci mają różnych sąsiadów. Jest to zgodne z wcześniejszą obserwacją dotyczącą niewielkiej liczby krawędzi grafu.

1.4. Wykreśl dendrogram i zaproponuj cięcie

Na podstawie wyników pochodzących z grupowania metodą Warda z metryką d1, otrzymanych w poprzednim podpunkcie wykreślono dendrogram.

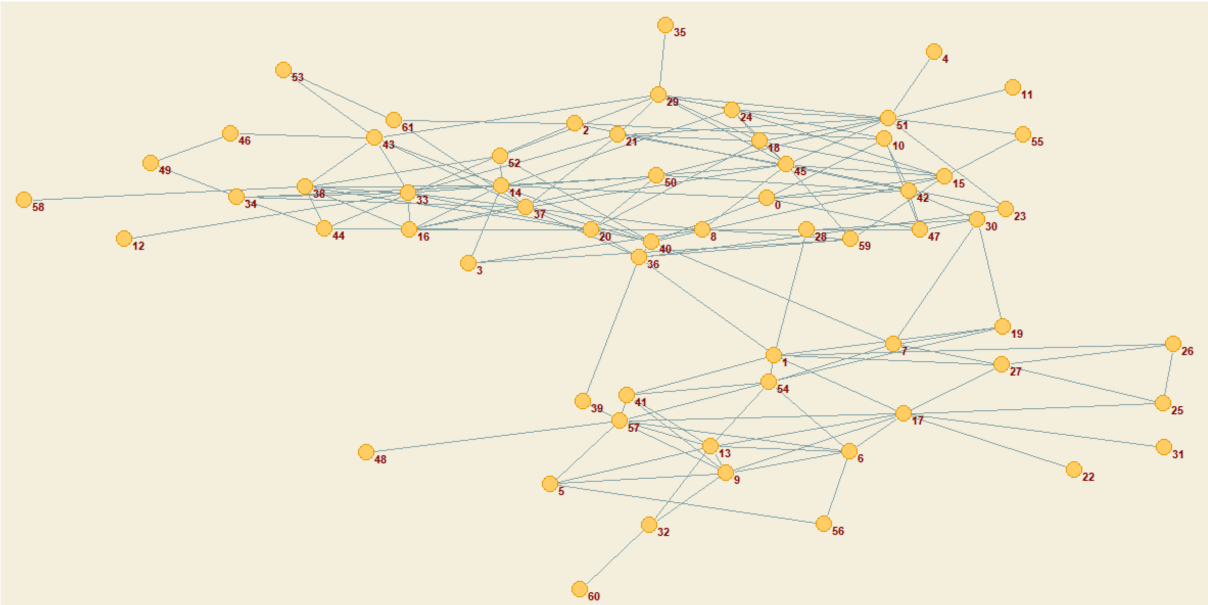


Rysunek 4. Dendrogram

Aby otrzymać najlepsze grupowanie można dokonać cięcia, które zaznaczono na dendrogramie jako niebieska linia. Pozwala ona na uzyskanie siedmiu klastrów o zbliżonych wartościach metryki.

1.5. Wykreśl wyodrębnione grupy

Możliwy jest także podział grup na dwa klastry co przedstawiono na rysunku 5. Widoczne jest, że istnieje niewielka liczba połączeń między węzłami należącymi do obydwu klastrów.



Rysunek 5. Wykreślone grupy

2. Zadanie B

283365 mod 6 = 3, zatem zbiór danych to „Sieć stron www”. Wszystkie zadania zostały wykonane przy użyciu języka Python w wersji 3.8.3. Cały kod wraz z opisem podpunktów, które realizuje został załączony na końcu pliku.

2.1. Zbadaj jaki jest rząd i rozmiar całej sieci: pierwotnej oraz po usunięciu pętli i duplikatów

W pierwszym etapie wczytano dane z pliku .txt do struktury *MultiGraph*, która znajdue się w bibliotece *networkx*. Następnie przy pomocy funkcji *number_of_nodes()* – odczytano rząd sieci, a przy pomocy funkcji *number_of_edges()* – rozmiar sieci. Otrzymano:

Tabela 2. Parametry pierwotnej sieci

Pierwotna sieć	
Rząd	**Rozmiar**
325729	1497134

Następnie usunięto duplikaty krawędzi poprzez transformację do struktury *Graph*. Otrzymano:

Tabela 3. Parametry sieci po usunięci duplikatów krawędzi

Sieć po usunięciu duplikatów krawędzi	
Rząd	**Rozmiar**
325729	1117563

Została usunięta ok. 1/4 krawędzi. Natomiast zgodnie z oczekiwaniami rząd sieci pozostał taki sam. W ostatnim etapie należało usunąć również pętle. Otrzymano:

Tabela 4. Parametry sieci po usunięciu duplikatów krawędzi oraz pętli

Sieć po usunięciu duplikatów krawędzi i pętli	
Rząd	**Rozmiar**
325729	1090108

Rozmiar sieci ponownie się zredukował.

2.2. Wyodrębnij największą składową spójną, zbadaj jej rząd i rozmiar

W tym podpunkcie przy użyciu funkcji *connected_components()* określono największą składową spójną sieci. Otrzymano:

Tabela 5. Parametry największej składowej spójnej sieci

Największa składowa spójna	
Rząd	**Rozmiar**
325729	1090108

Na podstawie powyższej tabeli można stwierdzić, że ani rozmiar, ani rząd sieci nie zmieniły się w stosunku do wartości w tabeli 4. Zatem można stwierdzić, że sieć od początku była spójna, czyli istniała ścieżka łącząca każdą parę wierzchołków. W dalszej części zadania B operowano na wyznaczonej największej składowej spójnej.

2.3. Wyznacz aproksymację średniej długości ścieżki, operując na próbie losowej 100, 1000, 10 tys. par wierzchołków

Początkowo należało wybrać próbę losową ze zbioru krawędzi. W tym celu losowano krawędzie, aż do momentu uzyskania zadanego rzędu sieci (100, 1000, 10 tys. par wierzchołków). Jednakże uzyskana w ten sposób sieć nie musi być spójna. Zatem następnym krokiem było wyodrębnienie największej składowej spójnej. Dla tak uzyskanej sieci przeprowadzono obliczanie średniej długości ścieżki. W przypadku 100 i 1000 wierzchołków obliczenia przeprowadzono 10-krotnie, a następnie wynik uśredniano. Uzyskano wyniki

```
Rzedy sieci spojnych:  [3, 4, 3, 3, 3, 4, 3, 3, 3, 3] dla liczby wezlow  100
Srednia dlugosc sciezki 1.3666666666666667
```

Rysunek 6. Średnia długość ścieżki dla próby losowej – 100 par wierzchołków.

```
Rzedy sieci spojnych:  [5, 6, 9, 8, 7, 8, 8, 5, 7, 8] dla liczby wezlow  1000
Srednia dlugosc sciezki 1.615873015873016
```

Rysunek 7. Średnia długość ścieżki dla próby losowej – 1000 par wierzchołków.

```
Rzedy sieci spojnych:  [147] dla liczby wezlow  10000
Srednia dlugosc sciezki 1.9176218432578511
```

Rysunek 8. Średnia długość ścieżki dla próby losowej – 10 tys. par wierzchołków.

Na podstawie uzyskanych wyników można stwierdzić, że średnia długość ścieżki wzrasta wraz ze zwiększeniem próby losowej. Jednakże warto zwrócić uwagę, na niewielkie rzędy uzyskanych sieci spójnych. W przypadku 100 węzłów wynoszą one 3-4. Oznacza to, że spośród 100 wylosowanych węzłów największa składowa spójna zawierała jedynie 3-4. Zatem średnia długość ścieżki obliczona na jej podstawie nie może być uznana za wiarygodną. W przypadku większych prób losowych rzędy sieci spójnych wzrastały, jednakże wciąż były niewielkie. Z tego względu należało by przeprowadzić obliczenia dla jeszcze większych prób losowych. Spośród uzyskanych wyników za najbardziej wiarygodny można uznać ten dla 10 tys. wierzchołków.

2.4. Wyznacz liczbę rdzeni o największym możliwym rzędzie, o drugim możliwe największym rzędzie, o trzecim możliwie największym rzędzie, jakie to są rzędy

W celu wyznaczenia rdzeni o największym możliwym rzędzie wykorzystano funkcję *core_number(),* która zwraca rząd każdego węzła. Następnie pogrupowano uzyskane wyniki. Uzyskano informację:

```
Trzy najwyzsze rzedy to 155 71 57
```

Rysunek 9. Największe rzędy sieci.

Następnie w celu poznania parametrów sieci o powyższych rzędach skorzystano z funkcji *k_core*. W celu weryfikacji czy otrzymane wyniki należą do tego samego k-rdzenia sprawdzono spójność sieci.

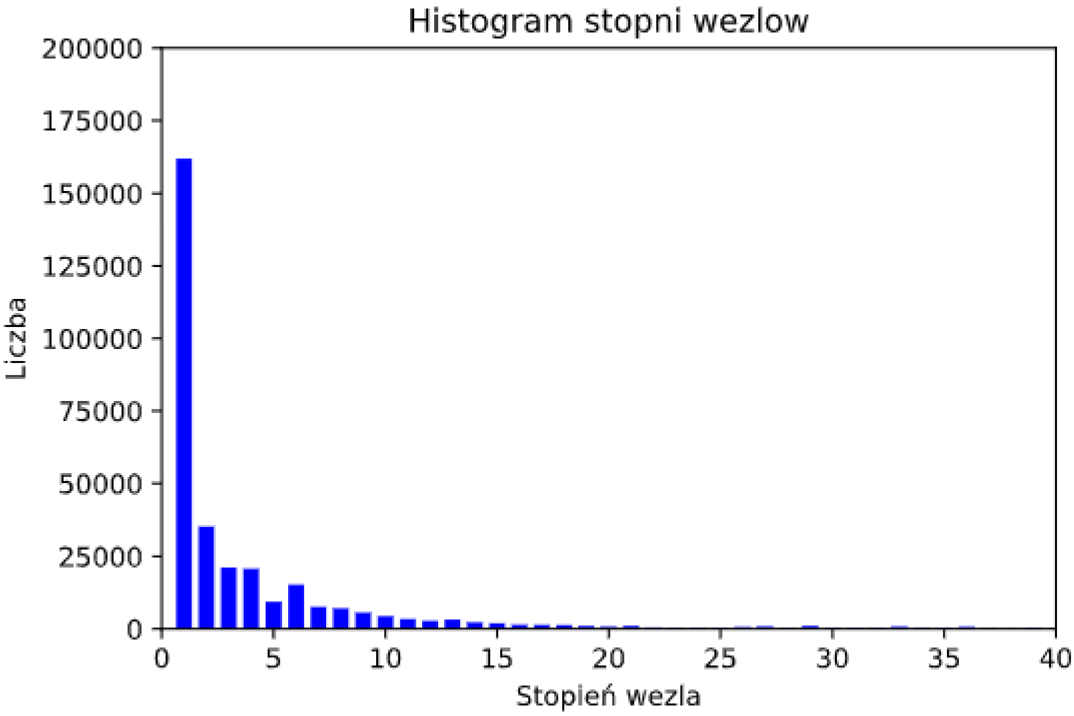
Tabela 6. Największe k-rdzenie sieci

k-rdzeń	Liczba sieci	Liczba wierzchołków	Liczba krawędzi
155	1	1367	107526
71	1	1961	129801
57	1	2429	144075

Na podstawie powyższej tabeli można stwierdzić, że największy k-rdzeń ma rozmiar 155, oznacza to że wszystkie należące do niego wierzchołki są co najmniej tego rzędu. Można by uznać, że drugi największy k-rdzeń ma rozmiar 154. Jednakże w przypadku analizowanej sieci, miałby on identyczną liczbę wierzchołków oraz krawędzi, jak największy rdzeń. Z tego powodu go pominięto. Drugi największy k-rdzeń ma wartość 71, a trzeci 57. Im mniejsza wartość k-rdzenia, tym większa liczba wierzchołków i krawędzi w nim zawarta.

2.5. Wykreśl rozkład stopni wierzchołków

Na podstawie liczebności wierzchołków każdego rzędu sporządzono histogram. Ze względu na fakt, że istnieją nawet pojedyncze węzły, które posiadają nawet do kilu tysięcy wychodzących z nich krawędzi, zdecydowano na przedstawienie histogramu jedynie dla zakresu 0 – 40. Dalsze wartości, stanowiące ogon wykresu zostały wykorzystane w dalszych podpunktach.

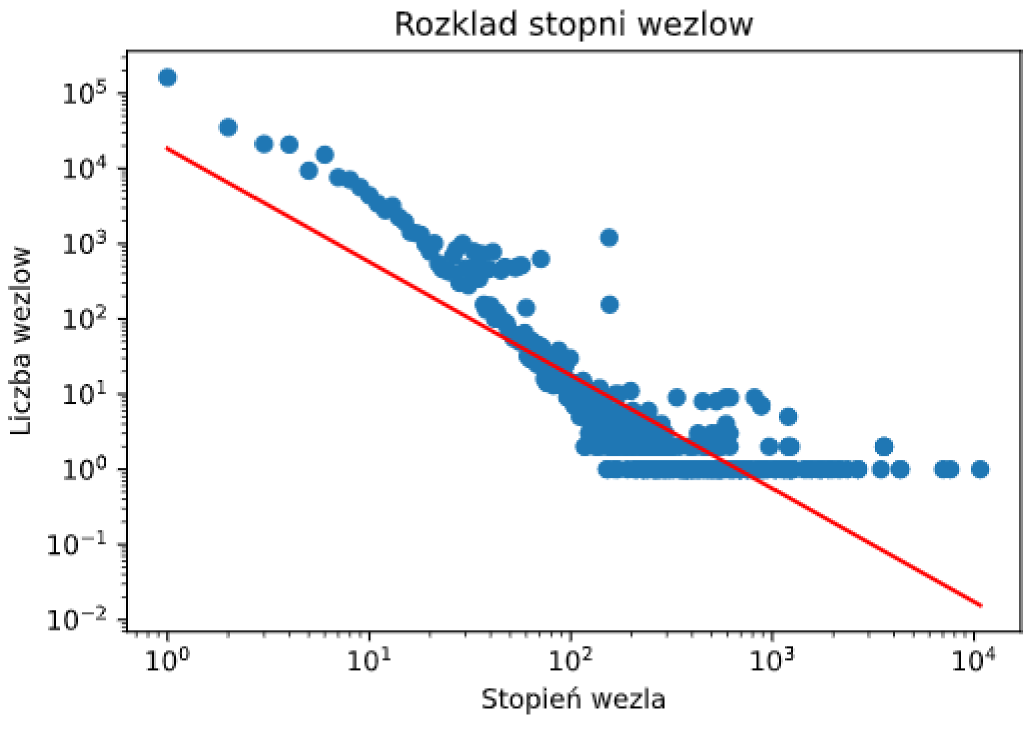


Rysunek 10. Rozkład stopni wierzchołków

Na podstawie histogramu można stwierdzić, że zdecydowanie przeważają wierzchołki o stopniu 1. Liczba wierzchołków o danym stopniu maleje wraz z jego wzrostem.

2.6. Wyznacz wykładnik rozkładu potęgowego metodą regresji dla dopełnienia dystrybuanty rozkładu stopni, dla przedziałów rozlokowanych logarytmicznie

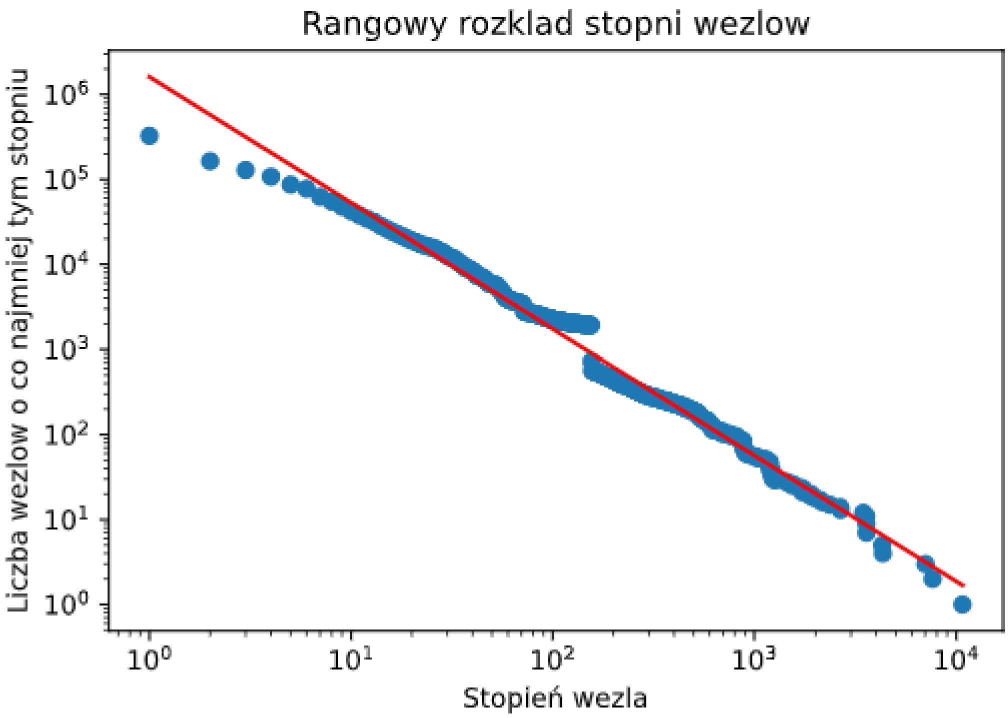
W pierwszym etapie wyznaczono rozkład stopni węzłów.



Rysunek 11. Rozkład stopni węzłów

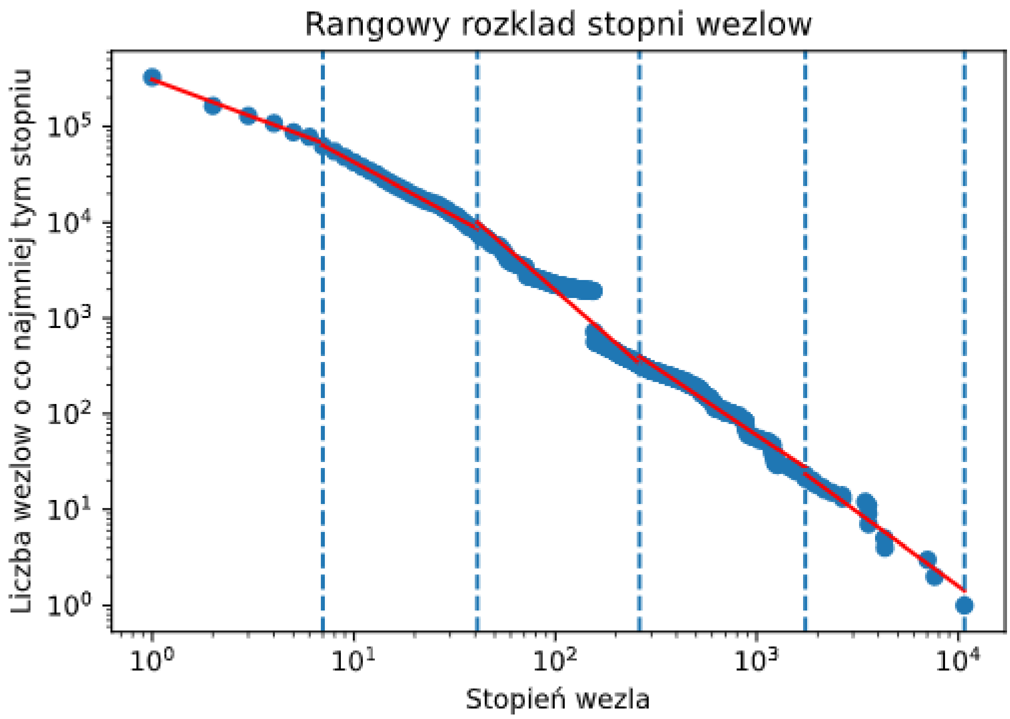
Widoczne jest, że w początkowej części wykresy punkty układają się liniowo. W dalszej części wykresu – odpowiadającej węzłom wysokiego stopnia pojawia się charakterystyczna chmura punktów o niewielkich wartościach liczby węzłów. Dla otrzymanego wykresu obliczono współczynnik kierunkowy prostej, który odpowiada wykładnikowi rozkładu potęgowego. Wyniósł on $a = -1{,}506$. Jednakże, ze względu na wspomniane przedziały prosta nie dopasowuje się w zadowalającym stopniu.

Następnym krokiem było sporządzenie wykresu rangowego, który odpowiada pojęciu dopełnienia dystrybuanty rozkładu do jedności. Otrzymano wykres:



Rysunek 12. Rankingowy rozkład stopni węzłów

Uzyskany wykres rangowy w skali podwójnie logarytmicznej układa się w dużej mierze liniowo. Współczynnik nachylenia wyniósł $a = -1{,}484$. Jednakże widoczne jest, że początkowy fragment wykresu jest nachylony w mniejszym stopniu w stosunku do jego końcowej części. Zatem podzielono wykres na 5 przedziałów rozlokowanych logarytmicznie. Uzyskane wyniki przedstawiono na wykresie.



Rysunek 13. Rangowy rozkład stopni węzłów z uwzględnieniem przedziałów

Na podstawie przedziałów odczytano wartości współczynnika nachylenia (wykładnika rozkładu potęgowego.

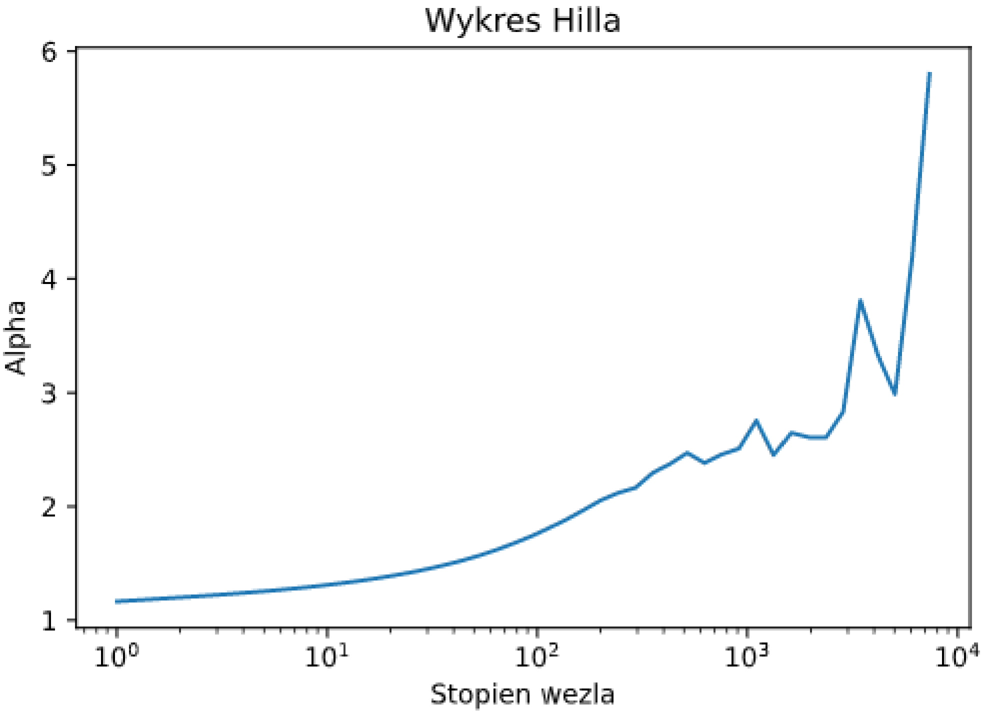
Tabela 7. Współczynniki nachylenia dla przedziałów

Nr przedziału	I	II	III	IV	V
Współczynnik nachylenia a	-0,784	-1,127	-1,837	-1,413	-1,53

Można zauważyć, że istnieje tendencja do spadku wartości współczynnika nachylenia dla wysokich stopni węzłów, co widoczne jest jako większy spadek funkcji. Ta zależność jest niewidoczna jedynie dla przedziału III. Jednakże analizując wykres rankingowy można zauważyć, że w tym przedziale istnieje duży skok funkcji, którego nie można zauważyć w pozostałych przedziałach.

2.7. Wyznacz wykres Hilla

W ostatnim etapie sporządzono wykres Hilla.



Rysunek 14. Wykres Hilla

Na podstawie wykresu można stwierdzić, że jeżeli chce się dopasować rozkład potęgowy dla całego zakresu, to skutkuje to uwzględnieniem głównie węzłów o niskim stopniu. Dla wyższych stopni nachylenie linii zaczyna wzrastać, co jest zgodne z wcześniejszymi obserwacjami.

Załącznik

```
# TASS - PROJEKT 1
# AUTOR: MATEUSZ HRYCIOW 283365

import networkx as nx
import numpy as np
from collections import Counter
import matplotlib.pyplot as plt
import powerlaw
import random

#1. Wczytanie grafu - "Siec stron www" oraz zbadanie rzedu oraz rozmiaru sieci
G = nx.read_edgelist('zadB_dane.txt', create_using=nx.MultiGraph)
print('Liczba wezlow i krawedzi pierwotnej sieci:')
print(G.number_of_nodes())
print(G.number_of_edges())

# Usuniecie petli i duplikatow krawedzi
G = nx.Graph(G)
print('Liczba wezlow i krawedzi po usunieciu duplikatow krawedzi:')
print(G.number_of_nodes())
print(G.number_of_edges())
G.remove_edges_from(nx.selfloop_edges(G))
print('Liczba wezlow i krawedzi po usunieciu petli i duplikatow:')
print(G.number_of_nodes())
print(G.number_of_edges())
```

```
#2. Wyodrebnienie najwiekszej skladowej spojnej
G_ss = max(nx.connected_components(G), key=len)
G_ss = G.subgraph(G_ss)
print('Liczba wezlow i krawedzi najwiekszej skladowej spojnej')
print(G_ss.number_of_nodes())
print(G_ss.number_of_edges())

#3. Wyznaczanie aproksymacji sredniej dlugosci sciezki
lengths = [100, 1000, 10000]
iters = 10
G_edges = G_ss.edges
for lens in (lengths):
    if lens == 10000:
        iters = 1
    connected_count = []
    paths_sum = 0
    for j in range(iters):
        print(j)
        G_len_edges = random.sample(G_ss.edges, int(lens/2))
        G_len = nx.Graph()
        G_len.add_edges_from(G_len_edges)
        while G_len.number_of_nodes() < lens:
            edge_add = random.sample(G_edges, int(lens/100))
            G_len.add_edges_from(edge_add)

        G_len_ss = max(nx.connected_components(G_len), key=len)
        connected_count.append(len(G_len_ss))
        G_len_ss = G_ss.subgraph(G_len_ss)
        shortest_path = nx.average_shortest_path_length(G_len_ss)
        paths_sum += shortest_path
    paths_avg = paths_sum/iters
    print("Rzedy sieci spojnych: ", connected_count, "dla liczby wezlow ", lens)
    print("Srednia dlugosc sciezki", paths_avg)

#4. Wyznaczanie liczby rdzeni o mozliwe najwiekszym rzedzie
vertex_degree = nx.core_number(G_ss)
vertex_degree_total = sorted(Counter(vertex_degree.values()).items())
print("Trzy najwyzsze rzedy to",vertex_degree_total[-1][0],vertex_degree_total[-2][0],vertex_degree_total[-3][0])

Core_1 = nx.k_core(G_ss, k = vertex_degree_total[-1][0])
print('Liczba wezlow i krawedzi k-rdzenia')
print(Core_1.number_of_nodes())
print(Core_1.number_of_edges())
Core_1_ss = max(nx.connected_components(Core_1), key=len)
Core_1_ss = G_ss.subgraph(Core_1_ss)
print('Liczba wezlow i krawedzi najwiekszej skladowej spojnej')
print(Core_1_ss.number_of_nodes())
print(Core_1_ss.number_of_edges())

Core_2 = nx.k_core(G_ss, k = vertex_degree_total[-2][0])
print('Liczba wezlow i krawedzi k-rdzenia')
print(Core_2.number_of_nodes())
print(Core_2.number_of_edges())
Core_2_ss = max(nx.connected_components(Core_2), key=len)
Core_2_ss = G_ss.subgraph(Core_2_ss)
print('Liczba wezlow i krawedzi najwiekszej skladowej spojnej')
print(Core_2_ss.number_of_nodes())
print(Core_2_ss.number_of_edges())

Core_3 = nx.k_core(G_ss, k = vertex_degree_total[-3][0])
print('Liczba wezlow i krawedzi k-rdzenia')
print(Core_3.number_of_nodes())
print(Core_3.number_of_edges())
Core_3_ss = max(nx.connected_components(Core_3), key=len)
Core_3_ss = G_ss.subgraph(Core_3_ss)
print('Liczba wezlow i krawedzi najwiekszej skladowej spojnej')
print(Core_3_ss.number_of_nodes())
print(Core_3_ss.number_of_edges())

#5. Wykreslanie rozkladu stopni wierzcholkow
```

```
degrees = G_ss.degree()
degrees = [deg[1] for deg in degrees]
degreeCount = Counter(degrees)
labels, values = zip(*degreeCount.items())

plt.bar(labels, values, color="b", width=0.7)
axes = plt.gca()
axes.set_xlim([0,40])
axes.set_ylim([0,200000])
plt.title("Histogram stopni wezlow")
plt.xlabel("Stopień wezla")
plt.ylabel("Liczba")
plt.show()

#6. Wyznaczanie wykladnika rozkladu potegowego

# Wykres liczby wezlow
degs = list(degreeCount.keys())
freqs = list(degreeCount.values())
freqs = [f for _,f in sorted(zip(degs,freqs))]
degs = sorted(degs)
fig = plt.figure()
ax = plt.gca()
ax.scatter(degs,freqs)
ax.set_yscale('log')
ax.set_xscale('log')
plt.title('Rozklad stopni wezlow')
plt.xlabel("Stopień wezla")
plt.ylabel("Liczba wezlow")
model = np.polyfit(np.log10(degs), np.log10(freqs), 1)
x_line = [min(degs), max(degs)]
y_line = [pow(10,model[1])*pow(x,model[0]) for x in x_line]
plt.plot(x_line, y_line, 'r')
plt.show()

# Wykres skumulowany
freqs_cum = []
for i in range(len(degs)):
    suma = 0
    for j in range (i, len(degs)):
        suma = suma + freqs[j]
    freqs_cum.append(suma)

fig = plt.figure()
ax = plt.gca()
ax.scatter(degs,freqs_cum)
ax.set_yscale('log')
ax.set_xscale('log')
plt.title('Rangowy rozklad stopni wezlow')
plt.xlabel("Stopień wezla")
plt.ylabel("Liczba wezlow o co najmniej tym stopniu")
model = np.polyfit(np.log10(degs), np.log10(freqs_cum), 1)
x_line = [min(degs), max(degs)]
y_line = [pow(10,model[1])*pow(x,model[0]) for x in x_line]
plt.plot(x_line, y_line, 'r')
plt.show()

przedzialy_N = 5
przedzialy = np.logspace(np.log10(min(degs)),np.log10(max(degs)),przedzialy_N+1)

fig = plt.figure()
ax = plt.gca()
ax.scatter(degs,freqs_cum)
ax.set_yscale('log')
ax.set_xscale('log')
plt.title('Rangowy rozklad stopni wezlow')
plt.xlabel("Stopień wezla")
plt.ylabel("Liczba wezlow o co najmniej tym stopniu")
```

```
fmin = 0
for i in range(1,przedzialy_N+1):

    deg_przedzial = [d for d in degs if d <= przedzialy[i]]
    fmax = degs.index(deg_przedzial[-1])
    freq_przedzial = freqs_cum[fmin:fmax+1]

    model = np.polyfit(np.log10(degs[fmin:fmax+1]), np.log10(freq_przedzial), 1)
    x_line = [degs[fmin], degs[fmax+1]]
    y_line = [pow(10,model[1])*pow(x,model[0]) for x in x_line]
    plt.plot(x_line, y_line, 'r')
    plt.axvline(degs[fmax+1], ls = "--")
    fmin = fmax + 1
    print("Nachylenie dla przedzialu ", i, " wynosi ", model[0])
plt.show()

#7. Wyzaczanie wykresu Hilla

NBINS = 50
bins = np.logspace(np.log10(min(degs)), np.log10(max(degs)), num = NBINS)
bcnt, bedge = np.histogram(np.array(degs), bins = bins)
alpha = np.zeros(len(bedge[:-2]))

for i in range(0, len(bedge)-2):
    fit = powerlaw.Fit(degs, xmin = bedge[i], discrete = True)
    alpha[i]=fit.alpha

plt.semilogx(bedge[:-2],alpha)
plt.title('Wykres Hilla')
plt.xlabel("Stopien wezla")
plt.ylabel("Alpha")
plt.show()
```