Sprawozdanie z projektu 1 – TASS 18Z

Sieć mała: 6. Współwystępowanie bohaterów Nędzników (wierzchołki – bohaterowie, krawędzie – współwystąpienie w rozdziale powieści)

Autor: Maciej Wiraszka, nr indeksu: 261215

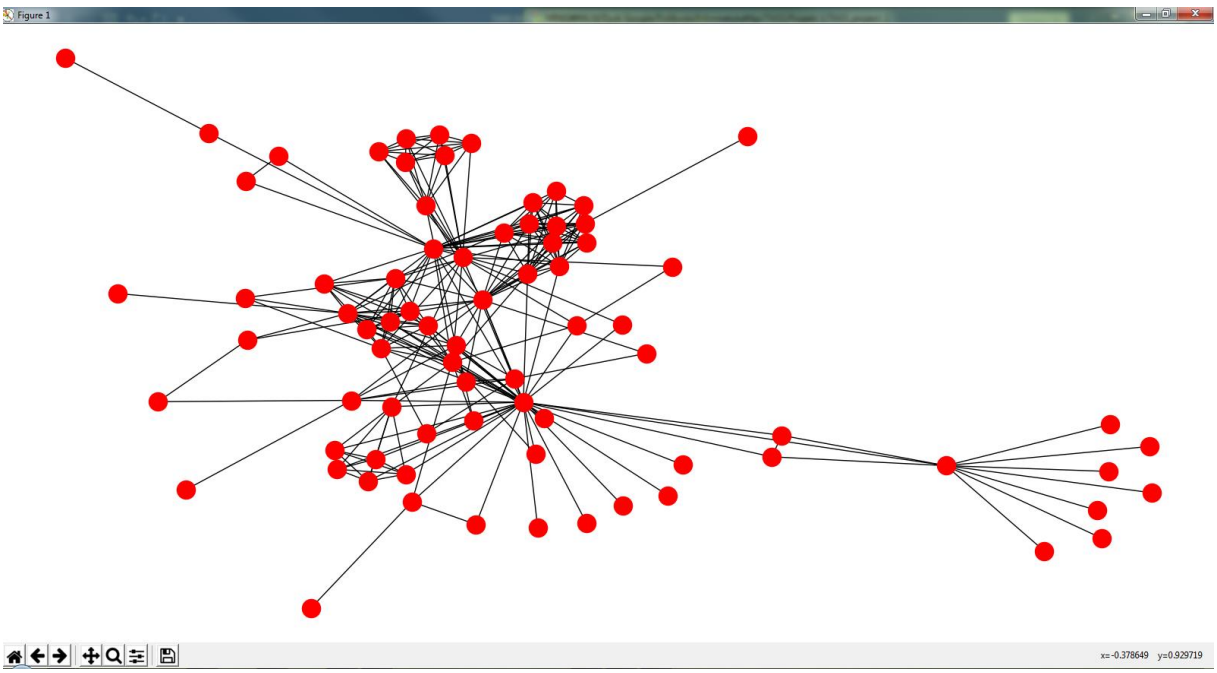
1. Analiza całego grafu

1.1. Skrypt *networkx_analysis.py* utworzony na potrzeby analizy

```python
import networkx as nx
import matplotlib.pyplot as plt
import time

def print_basic_graph_info(gr):
    print(f'Number of nodes:\t{gr.number_of_nodes()}')
    print(f'Number of edges:\t{gr.number_of_edges()}')

def main():
    print('\nLoading network from .gml file...')
    G_raw = nx.read_gml('network/lesmis.gml')   # read graph to networkx
    print('DONE\n')
    print_basic_graph_info(G_raw)

    print('\nRemoveing duplicated edges...')
    G = nx.Graph(G_raw)                         # delete duplicated edges
    print('DONE\n')
    print_basic_graph_info(G)

    print('\nConverting graph to undirected...')
    GU = G.to_undirected()                      # convert to undirected graph
    print_basic_graph_info(GU)
    print('DONE\n')

    print('\nConverting graph to .net file for Pajek usage...')
    nx.write_pajek(GU, 'network/lasmis.net')     # convert graph to .net file for Pajek usage
    print('DONE\n')

    print('\nGenerating connected components as subgraphs...')
    start = time.time() # start measuring time for the operation
    GU_connected_components = list(nx.connected_component_subgraphs(GU))         # generate connected components as subgraphs
    end = time.time()   # end measuring time for the operation
    print(f'DONE\t Generation time:\t{(end - start):.5f} seconds\n')

    print(f'\nNumber of connected components:\t{len(GU_connected_components)}\n') # count number of connected subgraphs

    print('\nExtracting largest connected component...')
    GUc = max(GU_connected_components, key=len) # extract largest connected component as subgraph
    print_basic_graph_info(GUc)
    print('DONE\n')

    print('\nPrinting largest connected component...')
    nx.draw(GUc)
    plt.show()
    print('DONE\n')

if __name__ == '__main__':
    main()
```

1.2. Wyniki wykonania skryptu *netwrkx_analysis.py*

```
Loading network from .gml file...
DONE

Number of nodes:        77
Number of edges:        254

Removeing duplicated edges...
DONE

Number of nodes:        77
Number of edges:        254

Converting graph to undirected...
Number of nodes:        77
Number of edges:        254
DONE


Converting graph to .net file for Pajek usage...
DONE


Generating connected components as subgraphs...
DONE     Generation time:       0.01560 seconds


Number of connected components: 1


Extracting largest connected component...
Number of nodes:        77
Number of edges:        254
DONE


Printing largest connected component...
DONE
```



1.3. Interpretacja wyników

Dane do grafu dla mojego tematu otrzymałem w formie pliku *.gml*, który wczytałem do skryptu za pomocą metody *read_gml* pakietu *networkx*.

Do usunięcia zduplikowanych krawędzi użyłem metody *Graph* pakietu *networkx*, podając jako argument wczytany graf. Na podstawie braku zmiany rozmiaru grafu po użyciu metody *Graph* wnioskuję, że w podanym grafie nie było zduplikowanych krawędzi.

Przekształcenia na graf nieskierowany dokonałem za pomocą metody *to_undirected* pakietu *networkx*.

Z racji niewspierania przez program *Pajek* rozszerzenia *.gml* dokonałem wyeksportowania grafu nieskierowanego do pliku *.net* za pomocą metody *write_pajek* pakietu *networkx*. Po eksporcie porównałem dane plików *.gml* oraz *.net* i stwierdziłem, że z punktu widzenia zadań do wykonania w projekcie są one spójne. Nie powiódł się jedynie eksport wag krawędzi (w *.net* wszystkie krawędzie miały wagę 1, gdy w *.gml* różne wagi), jednak nie ma to wpływu na dalszą analizę, dlatego postanowiłem tego nie poprawiać.

Do wyznaczenia składowych spójnych używając pakietu *networkx* użyłem metody *connected_component_subgraphs*. metoda wyznaczyła 1 składową spójną, którą dalej używam jako największą składową spójną analizowanego grafu. Wyznaczenie składowych spójnych zajęło około 0,0156 sekundy. Następnie zbadałem jej rząd (metodą *number_of_*nodes pakietu *networkx*, otrzymując wynik 77) i rozmiar (metodą *number_of_edges* pakietu *networkx*, otrzymując wynik 254). Aby zweryfikować powyższy wynik wyrysowałem graf używając pakietu *matplotlib*.

Aby wyznaczyć składowe spójne w programie *Pajek* załadowałem najpierw graf z wygenerowanego wcześniej pliku *.net* jako *Network*, a następnie użyłem opcji z menu głównego Network->Create Partition->Components->Weak, ustawiając 1 jako minimalną wielkość. Otrzymałem następujące wyniki, które zostały wygenerowane w mniej niż 1 sekundę:

```
================================================================================
Weak Components
================================================================================
 Working...
 Number of components: 1
 Size of the largest component: 77 vertices (100.000%).
 Time spent:  0:00:00
```

Powyższe wyniki są takie same, jak te wygenerowane przez przy użyciu pakietu *networkx*.

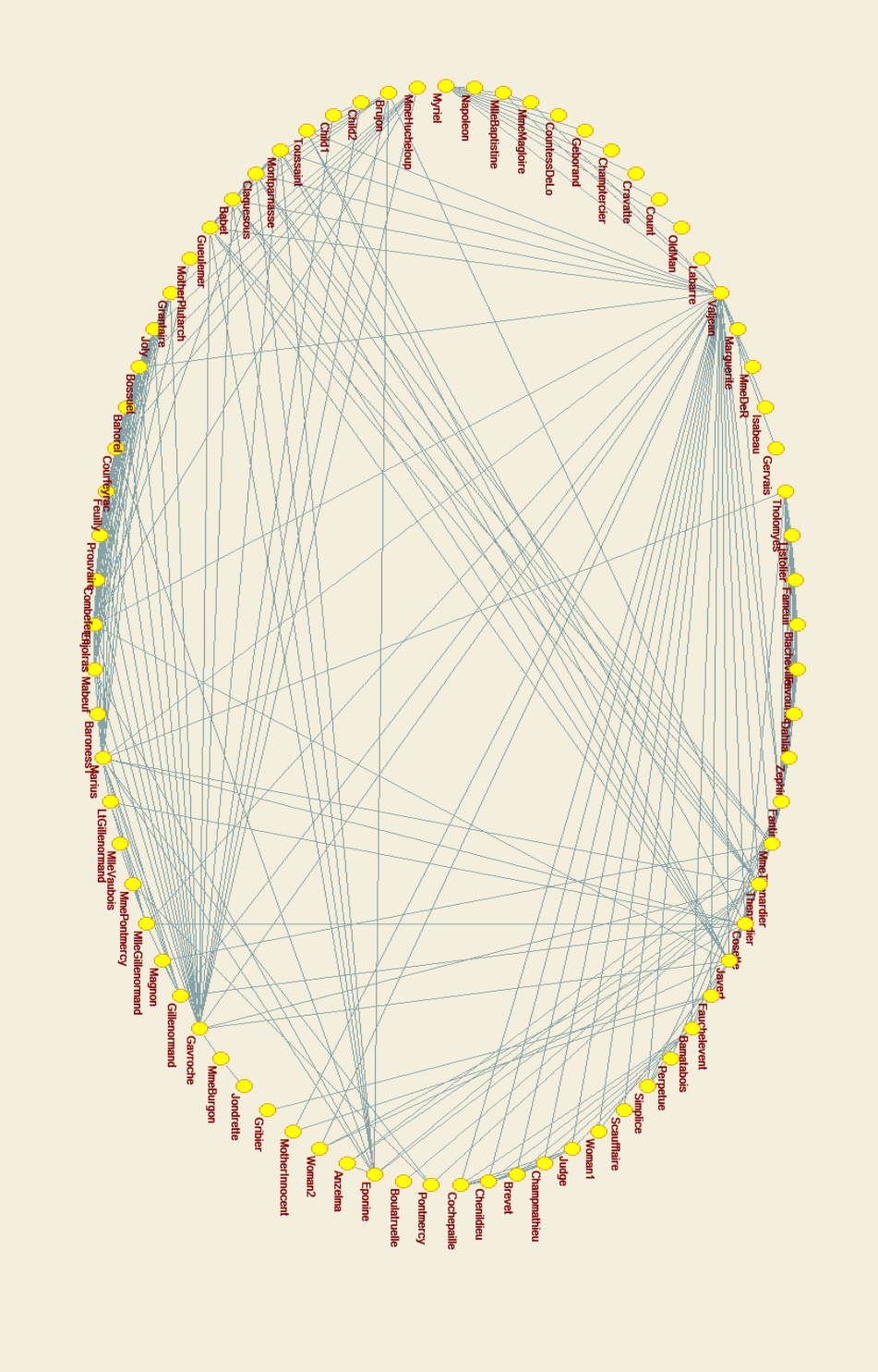
Podsumowując powyższe badania, analizowany graf jest spójny i posiada tylko jedną, słabo spójną składową będącą całym grafem. Różnicę czasu wykonania wyznaczania składowych spójnych przez różne narzędzia ciężko jest porównać, ze względu na specyfikę grafu (1 składowa spójna) oraz mniejszą dokładność podawania czasu wykonania przez *Pajeka*, jednak i tu i tu zajęło to poniżej 1 sekundy.

Analizując powyższe wyniki, doszedłem do wniosku, że w powieści Nędznicy wątki wszystkich postaci są ze sobą co najmniej pośrednio powiązane. Nie istnieją takie 2 grupy postaci, które byłyby od siebie odseparowane, czyli nie zachodziłaby relacja pomiędzy co najmniej 1 osobą w grupie pierwszej i 1 osobą w grupie drugiej (graf posiada tylko 1 składową spójną).

2. Szczegółowa analiza największej składowej spójnej (grafu)

2.1. Wykreślenie grafu

Kontynuując analizę w *Pajeku* z poprzedniego punktu, wykreśliłem graf (z menu głównego Draw->Network + First Partition) oraz zapisałem go jako mapę bitową (menu wykreślonego grafu Export->2D->Bitmap). Wyeksportowany obraz umieszczam poniżej:



2.2. Wyznaczanie 5 wierzchołków o największej wartości: bliskości, pośrednictwa i rangi

Największa wartość bliskości (menu główne Network->Create Vector->Centrality->Closeness->Input):

```
================================================================================
1. Input closeness centrality in N1 (77)
================================================================================
Dimension: 77
The lowest value:                       0.2568
The highest value:                      0.6441

Highest values:

      Rank    Vertex                     Value   Id
------------------------------------------------------
         1        12                    0.6441   Valjean
         2        56                    0.5315   Marius
         3        28                    0.5170   Javert
         4        26                    0.5170   Thenardier
         5        49                    0.5135   Gavroche
------------------------------------------------------
```

Największa wartość pośrednictwa (menu główne Network->Create Vector->Centrality->Betweenness):

```
================================================================================
2. Betweenness centrality in N1 (77)
================================================================================
Dimension: 77
The lowest value:                       0.0000
The highest value:                      0.5700

Highest values:

      Rank    Vertex                     Value   Id
------------------------------------------------------
         1        12                    0.5700   Valjean
         2         1                    0.1768   Myriel
         3        49                    0.1651   Gavroche
         4        56                    0.1320   Marius
         5        24                    0.1296   Fantine
------------------------------------------------------
```

Największa wartość rangi (menu główne Network->Create Vector->Centrality->Hubs-Authorities):

```
================================================================================
4. Hub Weights of N1 (77)
================================================================================
Dimension: 77
The lowest value:                       0.0022
The highest value:                      0.3178

Highest values:

      Rank    Vertex                     Value   Id
------------------------------------------------------
         1        49                    0.3178   Gavroche
         2        12                    0.2676   Valjean
         3        59                    0.2672   Enjolras
         4        56                    0.2591   Marius
         5        65                    0.2421   Bossuet
------------------------------------------------------
```

Analizując powyższe wyniki, dochodzę do wniosku, że główną postacią powieści jest Valjean.

2.3. Wyznaczanie największych klik

Największe kliki wyznaczyłem wyszukując kolejno grafy pełne (generowane z menu głównego Network->Create New Network->Complete Network->Directed) o coraz większym rzędzie w analizowanym grafie (ze względu na specyfikę obliczeń przekształconego do grafu skierowanego Network->Create New Network->Transform->Edges->Arcs), aż nie znaleziono żadnego grafu pełnego danego rzędu. Do tego celu używałem opcji z menu głównego Networks->Fragment(First in Second)).

Największe kliki w analizowanym grafie są rzędu 10, znajdują się 2 takie kliki.

```
 1.    1 - Gavroche
 2.    2 - Marius
 3.    3 - Mabeuf
 4.    4 - Enjolras
 5.    5 - Combeferre
 6.    7 - Feuilly
 7.    8 - Courfeyrac
 8.    9 - Bahorel
 9.   10 - Bossuet
10.   11 - Joly
```

```
 1.    1 - Gavroche
 2.    4 - Enjolras
 3.    5 - Combeferre
 4.    6 - Prouvaire
 5.    7 - Feuilly
 6.    8 - Courfeyrac
 7.    9 - Bahorel
 8.   10 - Bossuet
 9.   11 - Joly
10.   12 - Grantaire
```

Obliczenia tego punktu w Pajeku były bardzo czasochłonne. Aby przyśpieszyć obliczenia postanowiłem po każdym zwiększeniu rzędu wyszukiwanego grafu pełnego używać grafu znalezionego w ramach wyszukiwania poprzedniego (mniejszego o 1) rzędu.

Analizując powyższe wyniki dochodzę do wniosku, że w powieści powyższe postacie wchodzą w bardzo bliską, bezpośrednią interakcję między sobą, gdyż wszyscy wzajemnie się znają.

2.4. Wyznaczanie średnicy i średniej długości ścieżki w grafie

Do wyznaczenie średnicy oraz średniej długości ścieżki w grafie oraz średnicy użyłem opcji z głównego menu Network->Create Vector->Distribution of Distances.

```
================================================================================
Distribution of Distances
================================================================================
 Working...
 Number of unreachable pairs: 0
 Average distance among reachable pairs: 2.64115
 The most distant vertices: Napoleon (2) and MotherPlutarch (68). Distance is 5.
 Time spent:  00:00:00
```

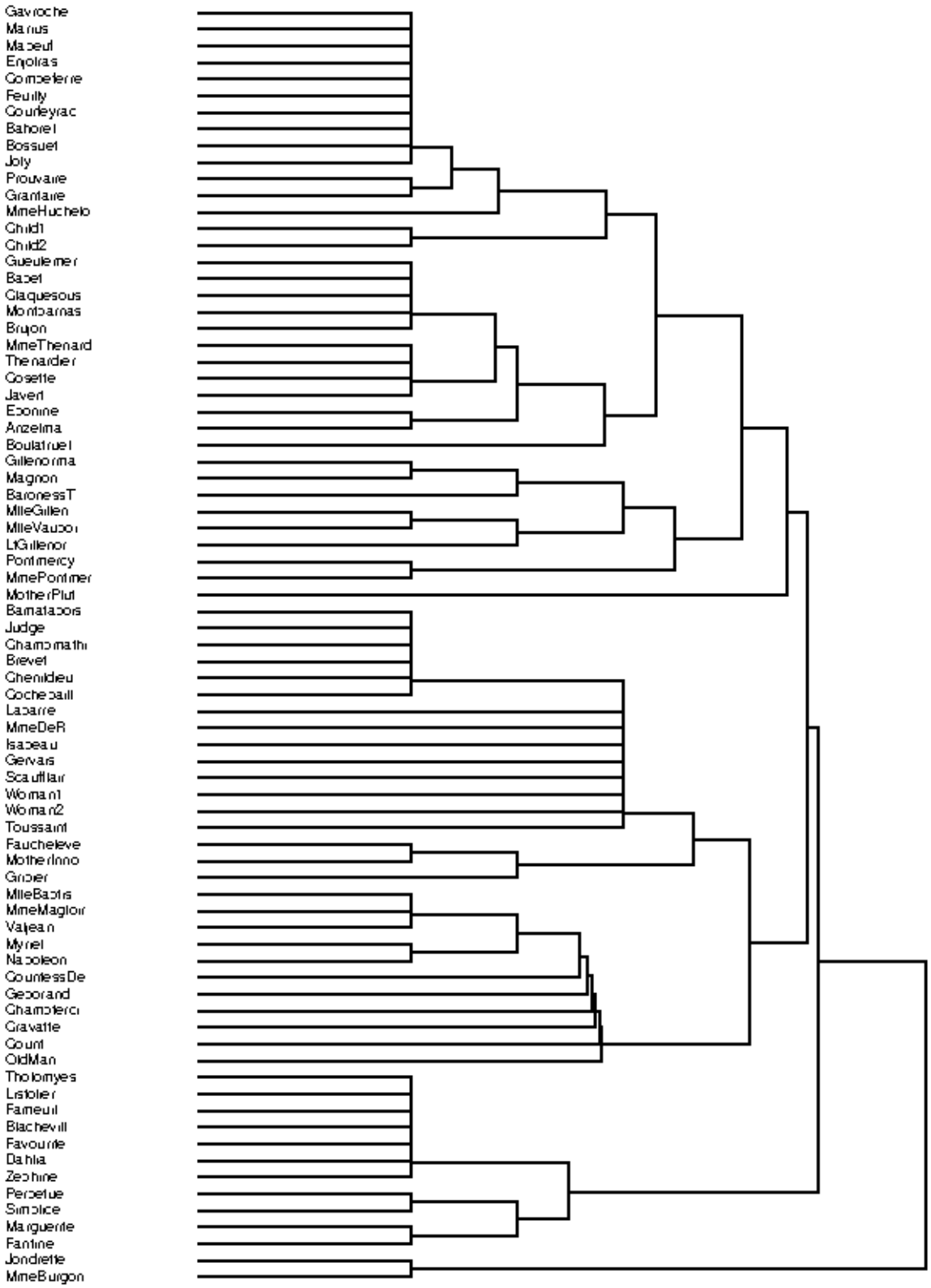
Wyznaczona średnica: 5

Wyznaczona średnia długość ścieżki: 2,64115

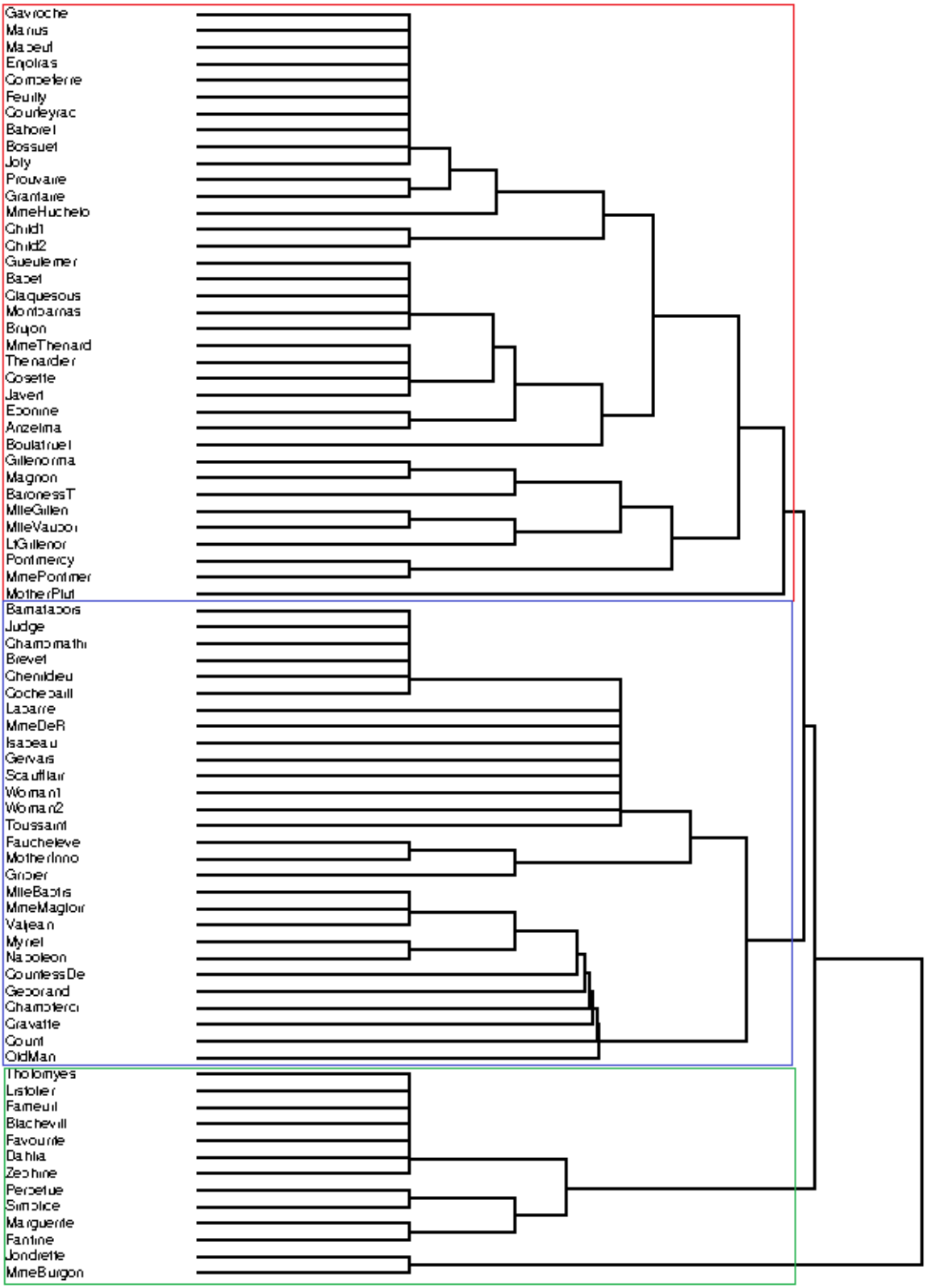
Analizując powyższe wyniki dochodzę do wniosku, że postacie Napoleon i Mother Plutarch są ze sobą w najbardziej odległej relacji. W ścieżce pomiędzy nimi, będącej średnicą grafu, jest 4 znajomych. Jednak jak na graf, w którym jest aż 77 wierzchołków uważam, że jest to dość mała odległość. Średnia długość ścieżki w grafie też jest dość niewielka, co jest spowodowane w znacznym stopniu dużą liczbą krawędzi występujących w grafie, a co za tym idzie, gęstymi stosunkami pomiędzy postaciami.

2.5. Grupowanie aglomeracyjne (identyfikator metody grupowania: UPGMA)

Do grupowania aglomeracyjnego z użyciem miary prawdopodobieństwa UPGMA zastosowałem opcję Network->Create Hierarchy->Clustering z wybranym Average jako Clustering Method, na macierzy najkrótszych ścieżek w zadanym grafie. Otrzymałem następujący dendrogram:



Powyższy dendrogram pokazuje grupowanie wierzchołków o największym podobieństwie średnich długości ścieżki do wszystkich innych wierzchołków. Poniżej zaproponowałem podział na grafu na 3 klastry:



Interpretując powyższe wyniki doszedłem do wniosku, że taki podział pozwoli na zgrupowanie bohaterów powieści, których średnie interakcje ze wszystkimi innymi bohaterami powieści są

podobne, czyli występują w wielu takich samych rozdziałach i/lub ich częstość występowania jest także podobna.