Лабораториска вежба 1
Вовед. Структура и својства на графови

NetworkX – Python библиотека за работа со графови

Граф во математиката и компјутерските науки претставува колекција од јазли кои можат да бидат поврзани помеѓу себе со врски (рабови). NetworkX претставува библиотека за програмскиот јазик Python за создавање, манипулација, и проучување на структурата, динамиката и функциите на сложените мрежи. Инсталацијата на NetworkX е преку следата команда:

```
pip install networkx
```

Главната структура е **`Graph`**. Еден граф може да биде насочен и ненасочен, и може да вклучува паралелни врски помеѓу два јазли, па соодветно во NetworkX се дефинирани:

- `Graph` – ненасочен едноставен граф (дозволува јамки (self-loops))
- `DiGraph` – насочен едноставен граф (дозволува јамки (self-loops))
- `MultiGraph` – ненасочен граф со повеќе врски помеѓу пар на јазли
- `MultiDiGraph` – насочен граф со повеќе врски помеѓу пар на јазли

За да се креира едноставен празен граф се користи податочната структура која што ни е потребна:

```
import networkx as nx

g = nx.Graph()
d = nx.DiGraph()
m = nx.MultiGraph()
h = nx.MultiDiGraph()
```

Вака креираните графови немаат дефинирани јазли и врски, па за да се креираат овие две компоненти се користат функциите **`nx.add_node(name, args*)`** и **`nx.add_edge(source, target, args*)`**. Врските во графот можат да бидат и тежински, па соодветно NetworkX поддржува поставување на тежини на врските:

```
g.add_edge(35, 2, weight=5)
```

Вчитување на податоци

Библиотеката вклучува поддршка за читање/запишување на графови во различен формат на датотеки. Пример, за читање на листа од врски може да користиме датотека во која секој ред е една врска опишана со идентификатор на стартниот јазел и дестинацискиот јазел, одделени со празно место:

```
g = networkx.read_edgelist("test.edges")
```

Друг пример е доколку читаме листа на соседство, односно датотека во која секој ред е даден јазел, по кој следуваат неговите соседи:

```
g = networkx.read_adjlist("test_adj.txt")
```

Дополнително, доколку имаме различен начин на претставување на графот кој се чита од дадена датотека, можеме датотеката да ја вчитаме во pandas Dataframe структура, од која потоа може да го креираме графот преку следната функција:

```
nx.from_pandas_edgelist(edgelist, edge_attr="label")
```

Освен дефинираните начини за креирање на граф структура, NetworkX е библиотека која содржи имплементации на многу функционалности и методи потребни за анализа и работа со мрежи. За подетални информации за работа со оваа библиотека погледнете ја документацијата.

Податочно множество Wikipedia vote network

Википедија е бесплатна енциклопедија, напишана во соработка со волонтери низ целиот свет. Мал дел од придонесувачите на Википедија се администратори, кои се корисници со пристап до дополнителни технички карактеристики што помагаат во одржувањето. Со цел корисникот да стане администратор, се издава Барање за администрација (RfA), а заедницата Википедија преку јавна дискусија или гласање одлучува кој да се промовира на администрација.

Мрежата ги содржи сите податоци за гласање на Википедија од почетокот на Википедија до јануари 2008 година. Јазлите во мрежата ги претставуваат корисниците на Википедија и насочен раб од јазол i до јазол j претставува дека корисникот i го изгласал корисникот j. Snap страната овозможува повеќе детали за овој граф.

Задачи

Задача 1 (5 поени)

Вчитајте го графот од податочното множество Wikipedia-Vote и анализирајте ја мрежата. Колкав е бројот на јазли и врски во мрежата?

Пресметајте ја и визуелизирајте ја дистрибуцијата на степени. Дистрибуцијата на степените јасно доловува само мала количина на информации за мрежата. Но, сепак таа информација дава важни индиции во структурата на мрежата. За да ги земеме степените на јазлите можеме да користиме:

```
[x[1] for x in list(nx.degree(graph))]
```

што ни дава степен на секој јазел, од кој понатаму може визуелно да се претстави хистограм, односно да се визуелизира дистрибуцијата на степените со помош на matplotlib:

```
import matplotlib.pyplot as plt

plt.hist([x[1] for x in list(nx.degree(graph))])
```

Визуелизирајте ја дистрибуцијата на логаритамска скала.

Поврзани компоненти (connected components) претставува метод со кој се извлекуваат посебните компоненти во графот кои не се поврзани со останатите делови. Со помош на имплементација на овој метод можеме да дознаеме дали мрежата која ја анализираме е целосно поврзана или се состои од засебни делови кои не се поврзани меѓусебно. Во NetworkX имплементацијата на овој метод се повикува на следниот начин:

```
nx.connected_components(g)
```

Пресметајте го бројот на поврзани компоненти и нивната големина (број на јазли). Колкава е големината на најголемата компонента? Пресметајте го дијаметарот на графот. Која е вредноста за просечниот кластерирачки коефициент?

Задача 2 (5 поени)

Во оваа задача креирајте два модели на графови со иста големина како графот од претходната задача:

- Random Graph `nx.gnp_random(n,p)`
- Small-World `nx.watts_strogatz_graph(n,k,p)`

Пресметајте ги следните барања и направете споредба помеѓу реалниот граф, и двата модели на графови:

1. Дистрибуција на степени
2. Поврзани компоненти
3. Дијаметар
4. Просечен кластерирачки коефициент