Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Владимирский государственный университет

имени Александра Григорьевича и Николая Григорьевича Столетовых»

(ВлГУ)

Колледж инновационных технологий и предпринимательства

**Лабораторная работа №3**

на тему: «Визуализация графов»

Вариант 21

Выполнила:

ст.гр. ИРсп-121

Орлова. С. В.

Проверил:

Ларин Е. С.

Владимир 2023

**Цель работы**

Изучить библиотеки и методы визуализации графов.

**Практическая часть**

**Задание для выполнения:**

1. Изучите представленные примеры программ визуализации графов.
2. Получить задания у преподавателя для построения графов:

**Граф 1**: список вершин, список ребер **не** взвешенного, **не** ориентированного графа.

**Граф 2**: матрицу весов взвешенного, ориентированного графа в котором **не** поддерживается отображение нескольких ребер между двумя вершинами, в котором есть петли.

**Граф 3**: матрицу весов взвешенного, ориентированного графа в котором поддерживается отображение нескольких ребер между двумя вершинами.

1. Перед написанием программы постройте граф в тетради и после этого составьте программы для отображения графов на экране. Не забываем установить нужные библиотеки в командной строке с помощью команды **pip install**.
2. В отчет добавьте коды программ. В кодах программ необходимо дописать недостающие комментарии к каждой строке кода.

**Задание 1.** создание первого графа - не взвешенного, не ориентированного.

Листинг кода:

import matplotlib.pyplot as plt #Импорт данного модуля для рисования графиков  
import networkx as nx # Импорт данного модуля для работы с графами  
G = nx.Graph() # создаём объект графа  
# определяем список узлов (ID узлов)  
nodes = [1, 2, 3, 4, 5, 6, 7]  
# определяем список рёбер  
# список кортежей, каждый из которых представляет ребро  
# кортеж (id\_1, id\_2) означает, что узлы id\_1 и id\_2 соединены  
  
edges = [(1, 2), (1, 3), (2, 3),(4,7), (3, 5), (5, 5), (3,6), (7,1), (7,2)]  
# добавляем информацию в объект графа  
G.add\_nodes\_from(nodes)  
G.add\_edges\_from(edges)  
# рисуем граф и отображаем его  
nx.draw(G, with\_labels=True, font\_weight='bold')  
plt.show()

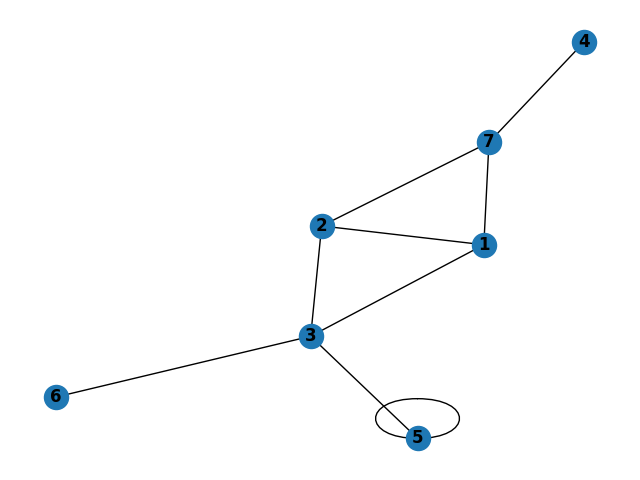


Рисунок 1 – Результат

**Задание 2.** Создание графа - взвешенного, ориентированного.

Листинг кода:

import networkx as nx # импортируем модуль networkx для работы с графами

import matplotlib.pyplot as plt # импортируем модуль matplotlib.pyplot для рисования графиков

PV = [[0.1,0.4,0.3,0.2],[0.873838,0.2,0.5,0.3],[0.5,0.3,0.5,0.5],[0,0,0.5,1],[0.7,0.2,0.1,0.1]] #определение матрицы весов

G = nx.DiGraph() # создание направленного графа

G.add\_nodes\_from(["s0", "s1", "s2","s3", "s4"]) # добавление узлов в граф

G.add\_edges\_from([("s0","s1"),

("s0","s3"),

("s1","s2"),

("s2","s3"),

("s2","s0"),

("s3","s4"),

("s3", "s1"),

("s0","s4"),

("s1", "s4"),

("s4", "s2")]) # добавление ребер в граф

edge\_loops = [("s0","s0"),

("s1","s1"),

("s2","s2"),

("s3","s3"),

("s4","s4")] # создание петель графа

pos = nx.shell\_layout(G) # определение позиций узлов графа для отображения

options = {

'node\_color': 'blue',

'node\_size': 500,

'width': 1,

'arrowstyle': '-|>',

'arrowsize': 20,

'edge\_color':'blue',

'font\_color':'white'

} # определение параметров для отображения ребер и узлов графа

options\_loops = {

'node\_size': 500,

'width': 1,

'arrowstyle': '-',

'arrowsize': 20,

'edge\_color':'blue'

} # определение параметров для отображения петель графа

nx.draw\_networkx\_edge\_labels(G, pos, edge\_labels={

("s0","s0"): (str(PV[0][0])+'\n'\*4),

("s1","s1"): (str(PV[1][1])+'\n'\*4),

("s2","s2"): (str(PV[2][2])+'\n'\*4),

("s3","s3"): (str(PV[3][3])+'\n'\*4),

("s4","s4"): (str(PV[4][2])+'\n'\*4)

}, font\_color='red', font\_size=6) # подписи ребер графа с вероятностями переходов

nx.draw(G, pos=pos, with\_labels=True, arrows=True, \*\*options) # создание графа с узлами и ребрами

nx.draw\_networkx\_edges(G, pos=pos, arrows=True, edgelist=edge\_loops, \*\*options\_loops) # созздание петель графа

nx.draw\_networkx\_edge\_labels(G, pos, edge\_labels={

("s0","s1"): str(PV[0][1]),

("s0","s3"): str(PV[0][3]),

("s1","s2"): str(PV[1][2]),

("s2","s3"): str(PV[2][3]),

("s2","s0"): str(PV[2][0]),

("s3","s4"): str(PV[3][2]),

("s3","s1"): str(PV[3][1]),

("s0","s4"): str(PV[0][2]),

("s1","s4"): str(PV[1][2]),

("s4","s2"): str(PV[4][2])

},

font\_color='red', label\_pos=(0.6)) # создание подписей ребер графа с вероятностями переходов для петель

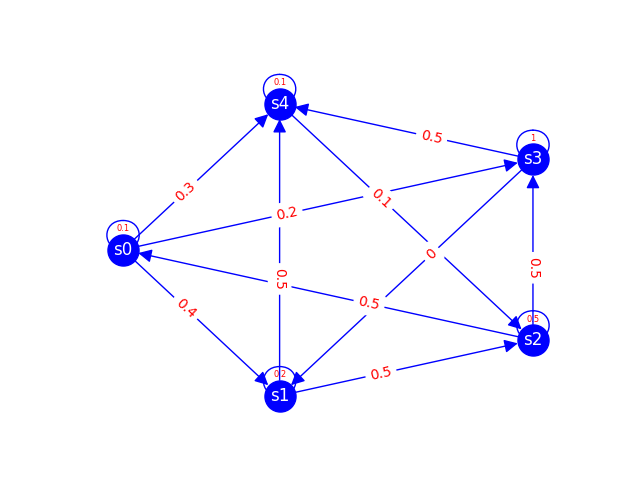
plt.show() # отображение графа

Рисунок 2 – Результат

**Задание 3.** Создание графа – взвешенного, ориентированного.

Листинг кода:

import networkx as nx # импорт данного модуля для работы с графами

import matplotlib.pyplot as plt # импорт данного модуля для рисования графиков

a = [[0,2,19,0],[0,0,0,1],[3,0,0,2],[1,10,3,4], [5,1,7,2],[5,1,7,2]] # определяем матрицу весов

G = nx.MultiDiGraph(directed=True) # создание направленного графа

G.add\_nodes\_from(["s0", "s1", "s2","s3", "s4"]) # добавление узлов в граф

G.add\_edges\_from([("s1","s3"),

("s2","s3"),

("s3","s4"),

("s4","s0"),

("s4","s1")]) # добавление ребер в граы

pos = nx.circular\_layout(G) # определение позиций узлов графа для отображения

options = {

'node\_color': 'blue',

'node\_size': 1000,

'width': 1,

'arrowstyle': '-|>',

'arrowsize': 20,

'edge\_color':'blue',

'font\_color':'white'

} # определение параметров для отображения узлов и ребер графа

options\_w = {

'node\_size': 1000,

'arrowstyle': '-|>',

'arrowsize': 20,

} # определение параметров для отображения ребер

nx.draw\_networkx(G, pos=pos,

with\_labels=True, arrows=True,

connectionstyle="arc3,rad=0.1",

\*\*options) # создание графа с узлами и ребрами

nx.draw\_networkx\_edge\_labels(G, pos, edge\_labels={

("s1","s3"): str(a[1][3]),

("s2","s3"): str(a[2][3]),

("s4", "s3"): str(a[4][3]),

("s4", "s0"): str(a[4][0]),

("s4", "s1"): str(a[4][1]),

}, font\_color='blue') # создание подписей ребер графа с вероятностями переходов

G.add\_edges\_from([("s0","s2"),("s2","s0")]) # добавление новых ребер в граф

nx.draw\_networkx\_edges(G, pos=pos, arrows=True,

edgelist=[("s0","s2")],

connectionstyle="arc3,rad=0.1", edge\_color='red',

\*\*options\_w) # создание новых ребер графа с красным цветом

nx.draw\_networkx\_edge\_labels(G, pos,

edge\_labels={("s0","s2"): str(a[0][2])},

label\_pos=(0.25), font\_color='red') # создание подписей новых ребер графа с вероятностями переходов

nx.draw\_networkx\_edges(G, pos=pos, arrows=True,

edgelist=[("s2","s0")],

connectionstyle="arc3,rad=0.1", edge\_color='green',

\*\*options\_w) # создание новых ребер графа с зеленым цветом

nx.draw\_networkx\_edge\_labels(G, pos,

edge\_labels={("s2","s0"): str(a[2][0])},

label\_pos=(0.25), font\_color='green') # создание подписей новых ребер графа с вероятностями переходов

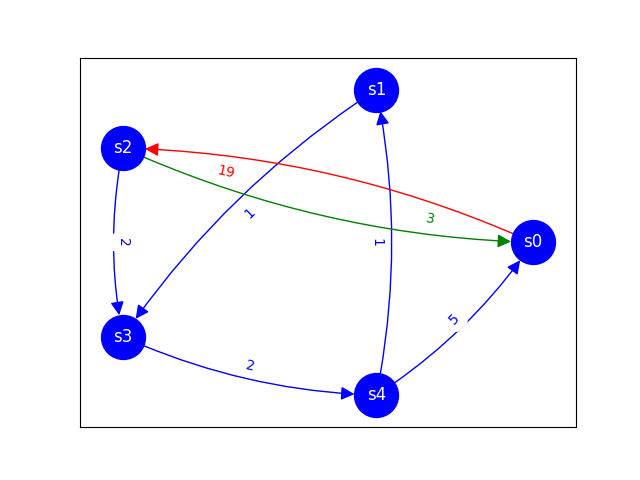
plt.show() # отображение графа

Рисунок 3 – Результат

**Вывод**

В ходе данной лабораторной работы изучены библиотеки и методы визуализации графов.