Министерство образования Республики Беларусь Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

КАФЕДРА ИНФОРМАТИКИ

Лабораторная работа №3

«Метод решения задачи о наидлиннейшем пути в направленном графе без контура»

Выполнил:

ст.гр.953503

Басенко К. А.

Проверил:

Дугинов О. И.

Минск 2022

# Постановка задачи

Пусть имеется направленный граф без контура (ориентированный ациклический граф). Требуется найти самый длинный путь между двумя вершинами в этом графе.

# Описание алгоритма метода

Задача о самом длинном пути — это задача поиска простого [пути](https://ru.wikipedia.org/wiki/%D0%9F%D1%83%D1%82%D1%8C_(%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2)) максимальной длины в заданном графе.Самый длинный путь A между двумя заданными вершинами s и t во взвешенном графе G — это то же самое, что и кратчайший путь в графе −G, полученном из G путём замены всех весов на веса с обратным знаком. Таким образом, если кратчайший путь можно найти в −G, то можно найти и самый длинный путь в G.

Для большинства графов такое преобразование бесполезно, поскольку создаёт циклы отрицательной длины в −G. Но

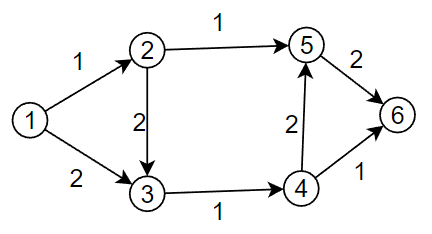
если G является [ориентированным ациклическим графом](https://ru.wikipedia.org/wiki/%D0%9D%D0%B0%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B0%D1%86%D0%B8%D0%BA%D0%BB%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D0%B3%D1%80%D0%B0%D1%84), невозможно создать отрицательный цикл и самый длинный путь в G может быть найден за [линейное время](https://ru.wikipedia.org/wiki/%D0%92%D1%80%D0%B5%D0%BC%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%BB%D0%BE%D0%B6%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%B0%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC%D0%B0), применив алгоритм поиска кратчайшего пути в −G (тоже ориентированный ациклический граф), который работает за линейное время.

Например, для любой вершины v в ориентированном ациклическом графе длина самого длинного пути, заканчивающегося в v, может быть получена выполнением следующих шагов:

* Осуществляем [топологическую сортировку](https://ru.wikipedia.org/wiki/%D0%A2%D0%BE%D0%BF%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%81%D0%BE%D1%80%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0) заданного ориентированного ациклического графа (ОАГ).
* Для каждой вершины v ОАГ в топологической сортировке вычисляем длину самого длинного пути, завершающегося в вершине v путём просмотра входящих дуг от соседей и добавления веса входяще дуги к максимальной длине в записях этих соседей. Если v не имеет входящих дуг, присваиваем длину самого длинного пути, кончающегося в v, нулю.

Когда это будет сделано, самый длинный путь во всём графе можно получить, начав с вершины v с самым большим записанным значением и проходя в обратном порядке, выбирая входящую дугу, у которой запись в начальной вершине имеет наибольшее значение.

# Результат работы Тест 1



Стартовая вершина: 1

Конечная вершина: 6

Результат:



**Код программы**

from math import inf

from tracemalloc import start

class Arc:

def \_\_init\_\_(self, node1, node2, cost):

self.node1 = node1

self.node2 = node2

self.cost = cost

def get\_topological\_order(graph: list, nodes: list):

color = ["white"] \* len(nodes)

result = []

def dfs(u: int):

color[u - 1] = "grey"

for i in graph:

if i.node1 == u:

if color[i.node2 - 1] == "white":

dfs(i.node2)

color[u - 1] = "black"

result.insert(0, u)

for i in nodes:

if color[i - 1] == "white":

dfs(i)

return result

def remove\_redudant\_nodes(ordered\_nodes: list):

is\_start = False

i = 0

while i < len(ordered\_nodes):

if ordered\_nodes[i] == start\_node:

is\_start = True

elif ordered\_nodes[i] == finish\_node:

if not is\_start:

return None

else:

return ordered\_nodes[0: i + 1]

elif not is\_start:

ordered\_nodes.remove(i)

i = i - 1

i = i + 1

def get\_longest\_path(nodes: list, graph: list, start\_node: list, finish\_node: list):

ordered\_nodes = get\_topological\_order(graph, nodes)

ordered\_nodes = remove\_redudant\_nodes(ordered\_nodes)

if ordered\_nodes == None:

print("No paths’s")

return None

opt = {}

x = {}

for i in ordered\_nodes:

if i == start\_node:

opt.update({i: 0})

x.update({i: None})

else:

max = -inf

previous\_node = 0

for arc in graph:

if arc.node2 == i:

value = opt[arc.node1] + arc.cost

if max < value:

max = value

previous\_node = arc.node1

opt.update({i: max})

x.update({i: previous\_node})

path = []

j = finish\_node

while True:

if j == start\_node:

path.insert(0, start\_node)

break

path.insert(0, j)

j = x[j]

return path

if \_\_name\_\_ == "\_\_main\_\_":

nodes = [1, 2, 3, 4, 5, 6]

graph = [

Arc(1, 2, 1), Arc(1, 3, 2),

Arc(2, 5, 1), Arc(3, 4, 1),

Arc(4, 5, 2), Arc(4, 6, 1),

Arc(5, 6, 2), Arc(2, 3, 2)

]

start\_node = 1

finish\_node = 6

path = get\_longest\_path(nodes, graph, start\_node, finish\_node)

print(f"Наидлинейший путь из {start\_node} в {finish\_node}: {path}")