Министерство цифрового развития, связи и массовых коммуникаций РФ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра ВС СибГУТИ

Отчёт

По курсовой работе

«Максимальный поток»

Выполнили студенты группы МГ-211:

Бурдуковский И.А.

Стояк Ю.К.

Проверил профессор кафедры ПМиК:

Рубан А.А.

Новосибирск 2023

**Оглавление**

[Задание 3](#_Toc157454975)

[Описание 4](#_Toc157454976)

[Результат работы программы 5](#_Toc157454977)

[Листинг 6](#_Toc157454978)

# Задание

Реализовать алгоритм Форда - Фалкерсона для решения задачи о максимальном потоке.

# Описание

Дан граф G(V, E) с пропускной способностью и потоком для рёбер из в v. Необходимо найти максимальный поток из источника s в сток t. На каждом шаге алгоритма действуют те же условия, что и для всех потоков:

* . Поток из в v не превосходит пропускной способности.
* .
* для всех узлов, кроме s и t. Поток не изменяется при прохождении через узел.

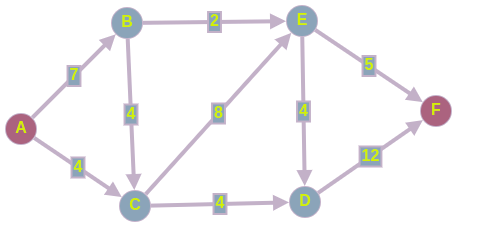
**Остаточная сеть**  — сеть с пропускной способностью и без потока.

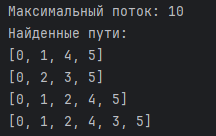
**Вход** Граф G с пропускной способностью C, источник s и сток t  
**Выход** Максимальный поток из s в t

1. для всех рёбер
2. Пока есть путь p из s в t в, такой что  для всех рёбер:
   1. Найти
   2. Для каждого ребра

Путь может быть найден, например, поиском в ширину (алгоритм Эдмондса — Карпа) или поиском в глубину/

# Результат работы программы





# Листинг

import pandas as pd  
  
  
class Graph:  
 def \_\_init\_\_(self, graph, source, sink):  
 self.graph = graph  
 self.vertex\_count = len(graph)  
 self.start = source  
 self.end = sink  
 self.pathes = []  
  
  
def find\_path\_BFS(g, parent):  
 visited = [False] \* g.vertex\_count  
 queue = [g.start]  
 visited[g.start] = True  
  
 while queue:  
 u = queue.pop(0)  
 # print(g.graph[u])  
 for ind, val in enumerate(g.graph[u]):  
 if visited[ind] == False and val > 0:  
 queue.append(ind)  
 visited[ind] = True  
 parent[ind] = u  
  
 if visited[g.end]:  
 return True  
 else:  
 return False  
  
  
def ford\_fulkerson(g):  
 parent = [-1] \* g.vertex\_count  
 max\_flow = 0  
  
 while find\_path\_BFS(g, parent):  
  
 path\_history = [g.end]  
  
 path\_flow = float("Inf")  
 s = g.end  
 while s != source:  
 path\_flow = min(path\_flow, g.graph[parent[s]][s])  
 s = parent[s]  
 path\_history = [s] + path\_history  
 # print(s)  
  
 g.pathes.append(path\_history)  
  
 max\_flow += path\_flow  
  
 v = g.end  
 while (v != source):  
 u = parent[v]  
 g.graph[u][v] -= path\_flow  
 g.graph[v][u] += path\_flow  
 v = parent[v]  
  
 return max\_flow  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 graph = []  
  
 with open('ford\_falkerson\_flow3.in', 'r') as file:  
 for line in file.readlines():  
 rows = list(map(int, line.split()))  
 graph.append(rows)  
  
 print("Граф:")  
 print(pd.DataFrame(graph), "\n")  
  
 source = 0  
 sink = 5   
  
 g = Graph(graph, source, sink)  
  
 print("Максимальный поток: %d " % ford\_fulkerson(g))  
 print("Найденные пути: ")  
 for path in g.pathes:  
 print(path)  
  
 print("Граф:")  
 print(pd.DataFrame(graph), "\n")