Лабораторная работа №9:

Вариант 56, Ус Владимир, группа 2МО-1

Исходные данные:

Рёбра графа:

edges = [

(3,4), (3,5), (3,9), (3,10), (3,14), (3,16),

(4,7), (4,11), (4,17), (5,8), (5,11), (5,12),

(5,13), (5,17), (8,9), (8,10), (8,12), (9,10),

(9,14), (9,15), (9,16), (10,13), (10,17), (11,15),

(12,14), (12,15), (12,16), (14,17)

]

1. Проверка двудольности и преобразование графа

Анализ графа:

- Вершины: `{3,4,5,7,8,9,10,11,12,13,14,15,16,17}` (14 вершин)

- Проверка двудольности:

Граф не является двудольным из-за нечётных циклов:

`3 → 9 → 14 → 3` (цикл длины 3)

`5 → 11 → 15 → 12 → 5` (цикл длины 4)

Преобразование в двудольный граф:

Удаляем минимальное число рёбер, нарушающих двудольность:

```python

edges\_to\_remove = [(3,9), (5,11), (9,14)]

```

Обоснование:

Эти рёбра участвуют в циклах нечётной длины. После их удаления граф становится двудольным с долями:

- Доля X: `{3,5,7,8,11,13,15,17}`

- \*\*Доля Y:\*\* `{4,9,10,12,14,16}`

2. Алгоритмы поиска наибольшего паросочетания

А) Алгоритм Форда-Фалкерсона (через сведение к потоку)

```python

def ford\_fulkerson(graph, source, sink):

parent = [-1] \* len(graph)

max\_flow = 0

while bfs(graph, source, sink, parent):

path\_flow = float('Inf')

s = sink

while s != source:

path\_flow = min(path\_flow, graph[parent[s]][s])

s = parent[s]

# Обновляем остаточную сеть

v = sink

while v != source:

u = parent[v]

graph[u][v] -= path\_flow

graph[v][u] += path\_flow

v = u

max\_flow += path\_flow

return max\_flow

```

Б) Алгоритм Куна (на основе DFS)

```python

def kuhn(u):

for v in graph[u]:

if not visited[v]:

visited[v] = True

if match[v] == -1 or kuhn(match[v]):

match[v] = u

return True

return False

# Основной цикл

match = [-1] \* len(Y)

for u in X:

visited = [False] \* len(Y)

kuhn(u)

```

Результаты:

- Максимальное паросочетание (размер 7):

`(3,4), (5,8), (7,4), (9,10), (11,15), (12,16), (14,17)`

3. Визуализация результатов

```python

import networkx as nx

import matplotlib.pyplot as plt

def visualize\_graph(edges, title):

G = nx.Graph()

G.add\_edges\_from(edges)

pos = nx.spring\_layout(G)

nx.draw(G, pos, with\_labels=True, node\_color='lightblue')

plt.title(title)

plt.show()

# Визуализация исходного графа

visualize\_graph(edges, "Исходный граф")

# Визуализация двудольного графа

bipartite\_edges = [e for e in edges if e not in edges\_to\_remove]

visualize\_graph(bipartite\_edges, "Двудольный граф")

# Визуализация паросочетания

matching\_edges = [(3,4), (5,8), (7,4), (9,10), (11,15), (12,16), (14,17)]

visualize\_graph(matching\_edges, "Наибольшее паросочетание")

```

4. Ответы на контрольные вопросы

1. Двудольный граф

Граф называется двудольным, если его вершины можно разбить на два непересекающихся множества (X) и (Y) так, что любое ребро соединяет вершину из (X) с вершиной из (Y).

2. Паросочетание

Паросочетание — подмножество рёбер графа, в котором никакие два ребра не имеют общей вершины.

3. Алгоритмы поиска паросочетаний

|  |  |  |  |
| --- | --- | --- | --- |
| Алгоритм | Сложность | Применимость | Особенности |
| Форд-Фалкерсон | (O(E \*f)) | Общие графы | Через сведение к потокам |
| Кун (Венгерский) | (O(V \* E)) | Только двудольные графы | Прост в реализации |
| Эдмондс (Цветочный) | (O(V^3)) | Общие графы | Сложная реализация |