Algorytmy do zestawu nr 5[1]

Problem maksymalnego przepływu

- G – sieć przepływowa, dla której szukamy maksymalnego przepływu,
- s – źródło sieci przepływowej,
- t – ujście sieci przepływowej,
- (u, v) – krawędź (kanał) z wierzchołka u do wierzchołka v,
- $c(u, v)$ – przepustowość krawędzi (u, v),
- $f(u, v)$ – przepływ krawędzi (u, v),
- G_f – sieć rezydualna dla G (z łac. *residuum* – reszta, pozostałość),
- $c_f(u, v)$ – przepustowość rezydualna krawędzi (u, v):

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{jeśli } (u, v) \text{ należy do } G, \\ f(v, u) & \text{jeśli } (v, u) \text{ należy do } G, \\ 0 & \text{w przeciwnym wypadku;} \end{cases} \tag{1}$$

- p – ścieżka rozszerzająca w G_f,
- $c_f(p)$ – przepustowość rezydualna ścieżki p (najmniejsza przepustowość rezydualna krawędzi tworzących p).

1. Algorytm Forda-Fulkersona

Algorytm 1: `ford_fulkerson`(G, s, t)

```
for każda krawędź (u, v) należąca do grafu G do
    f(u, v) ← 0                              ▷ Zerowanie przepływów dla wszystkich krawędzi
end for
while istnieje ścieżka rozszerzająca p z s do t w sieci rezydualnej G_f do
    c_f(p) ← min{c_f(u, v) dla wszystkich krawędzi (u, v) ∈ p}
    for każda krawędź (u, v) ∈ p do
        if krawędź (u, v) należy do grafu G then
            f(u, v) ← f(u, v) + c_f(p)       ▷ Zwiększamy przepływ przez krawędź (u, v)
        else
            f(v, u) ← f(v, u) − c_f(p)       ▷ Przypadek kasowania przepływu
        end if
    end for
end while
```

[1] Na podstawie: Cormen Thomas H., Leiserson Charles E., Rivest Ronald L., *Wprowadzenie do algorytmów*, Wyd. 4, Warszawa, Wydawnictwo Naukowo – Techniczne, 2001, ISBN 83-204-2665-0.

Algorytm Forda-Fulkersona można implementować, wybierając ścieżki rozszerzające na różne sposoby. Jedną z jego wersji jest algorytm Edmondsa-Karpa, w którym w pętli `while` zawsze wybieramy ścieżkę p o najmniejszej liczbie krawędzi (przy pomocy przeszukiwania wszerz – algorytm przedstawiono poniżej).

2. Przeszukiwanie wszerz

- d_s – tablica odległości: $d_s[v]$ – długość najkrótszej ścieżki między wierzchołkiem startowym s a wierzchołkiem v mierzona jako liczba krawędzi,
- p_s – tablica poprzedników w najkrótszych ścieżkach.

Algorytm 2: BFS(G, s) ▷ *s - wierzchołek startowy w grafie G*

```
init(G, s)
Utwórz pustą kolejkę Q
Dodaj s do kolejki Q
while Q ≠ ∅ do                                     ▷ Dopóki kolejka nie jest pusta
    Ściągnij wierzchołek z początku kolejki i przypisz go do v
    for każdy wierzchołek u ∈ G będący sąsiadem v do
        if d_s[u] = ∞ then
            d_s[u] ← d_s[v] + 1
            p_s[u] ← v
            Dodaj u do kolejki Q
        end if
    end for
end while
```

W algorytmie Forda-Fulkersona potrzebujemy znaleźć najkrótszą ścieżkę ze źródła s do ujścia t w grafie G_f. Dlatego wywołanie algorytmu BFS(G_f, s) można zakończyć w momencie znalezienia ścieżki z s do t – algorytm nie zawsze będzie musiał przechodzić przez cały graf.