ALGORYTMY GRAFOWE

dr hab. Mariusz Meszka

Akademia Górniczo-Hutnicza w Krakowie

http://home.agh.edu.pl/∼meszka

Program wykładu

1. Maksymalne przepływy w sieciach.
2. Inne zagadnienia przepływowe: sieci z dolnym ograniczeniem przepływu, przepływy o minimalnym koszcie, przepływy uogólnione.
3. Skojarzenia w grafach.
4. Grafy planarne – testowanie planarności oraz znajdowanie planarnej reprezentacji.
5. Kolorowanie wierzchołkowe oraz krawędziowe.
6. Pokrycie wierzchołkowe, dominowanie, kliki i zbiory niezależne.
7. Grafy przekątniowe i przedziałowe – rozpoznawanie.
8. Podziały i dekompozycje grafów.

Literatura

[1] S. Even, Graph Algorithms 2nd Edition, Cambridge University Press 2011.
[2] W.L. Kocay, D.L. Kreher, Graphs, Algorithms and Optimization 2nd Edition, CRC Press 2017.
[3] K. Erciyes, Guide to Graph Algorithms: Sequential, Parallel and Distributed, Springer 2018.
[4] D. Williamson, Network Flow Algorithms, Cambridge University Press 2019.

Siecią $G = (V, A, s, t, c)$ nazywamy graf skierowany (V, A), w którym dwa wierzchołki s, t (nazywane odpowiednio *źródłem* i *ujściem*) są wyróżnione, a c jest funkcją przepustowości

$$c: \ A \mapsto \mathbb{R}_+.$$

Przepływem w sieci G nazywamy funkcję

$$f: \ A \mapsto \mathbb{R}_+$$

spełniającą warunki:
(1) $\forall (u, v) \in A: \quad 0 \leqslant f(u, v) \leqslant c(u, v)$
(2) $\forall v \in V \setminus \{s, t\}:$

$$\sum_{u \in V:\ (u,v) \in A} f(u, v) = \sum_{u \in V:\ (v,u) \in A} f(v, u).$$

Wartością przepływu f jest liczba
$|f| = \sum_{(s,u) \in A} f(s,u) = \sum_{(u,t) \in A} f(u,t)$.

Problem maksymalnego przepływu

Wejście: Sieć $G = (V, A, s, t, c)$
Wyjście: Przepływ f_{max} :

$$|f_{max}| = \max\{|f| : \ f \text{ jest przepływem w } G\}.$$

Siecią residualną dla sieci $G = (V, A, s, t, c)$ oraz przepływu f nazywamy sieć $G_f = (V, A_f, s, t, c_f)$, w której $A_f = A \cup \{(u, v) : \; f(v, u) > 0\}$ oraz:

$$c_f(u, v) = \begin{cases} c(u, v) - f(u, v) & \text{jeśli } (u, v) \in A \\ f(v, u) & \text{wpp} \end{cases}$$

Ścieżką powiększającą dla przepływu f jest ścieżka od s do t w sieci residualnej G_f.

Niech p będzię ścieżką powiększającą dla przepływu f. *Przepustowość residualna* ścieżki p wynosi $c_f(p) = \min\{c_f(u, v) : \; (u, v) \in p\}$.

Metoda Forda-Fulkersona

for każdy łuk $(u, v) \in A$ **do**
    $f(u, v) := 0$
$G_f := G$
while istnieje ścieżka powiększająca p w G_f **do**
    $c_f(p) := \min\{c_f(u, v) : (u, v) \in p\}$
    for każdy łuk $(u, v) \in p$ **do**
        $f(u, v) := f(u, v) + c_f(p)$
        $c_f(u, v) := c(u, v) - f(u, v)$
        $c_f(v, u) := f(u, v)$

Jeśli c jest funkcją całkowitoliczbową to $T = O(|A||c|)$.

Algorytm Edmondsa-Karpa

Do znajdowania ścieżek powiększających w metodzie Forda-Fulkersona wykorzystawane jest przeszukiwanie wszerz (BFS).

$$T = O(|V||A|^2)$$

Przekrojem w sieci $G = (V, A, s, t, c)$ nazywamy taki podział (S, T) zbioru V, w którym $s \in S$ oraz $t \in T$.

Przepustowość przekroju
$c(S, T) = \sum_{(u,v) \in A: u \in S, v \in T} c(u, v)$
Przepływ przez przekrój
$f(S, T) = \sum_{(u,v) \in A: u \in S, v \in T} f(u, v)$

Twierdzenie o maksymalnym przepływie i minimalnym przekroju

Niech f będzie przepływem w sieci $G = (V, A, s, t, c)$.
Następujące warunki są równoważne.
(1) Przepływ f w G jest maksymalny
(2) Sieć residualna G_f nie zawiera ścieżki powiększającej
(3) Istnieje przekrój (S, T), dla którego zachodzi $|f| = c(S, T)$.

Przedprzepływem w sieci G nazywamy funkcję

$$g: \ A \mapsto \mathbb{R}_+$$

spełniającą warunki:
(1) $\forall (u,v) \in A: \ \ 0 \leqslant g(u,v) \leqslant c(u,v)$
(2) $\forall v \in V \setminus \{s,t\}:$

$$\sum_{u \in V\colon (u,v) \in A} g(u,v) - \sum_{u \in V\colon (v,u) \in A} g(v,u) \geqslant 0.$$

Nadmiarem w wierzchołku v nazywamy liczbę

$$e(v) = \sum_{u \in V\colon (u,v) \in A} g(u,v) - \sum_{u \in V\colon (v,u) \in A} g(v,u).$$

Wierzchołek $v \in V \setminus \{s,t\}$ jest *nadmiarowy* jeśli $e(v) > 0$.

Funkcję $h: V \mapsto \mathbb{N}$ nazywamy *funkcją wysokości* jeśli
(1) $h(s) = |V|$
(2) $h(t) = 0$
(3) $\forall (u, v) \in A_g: \quad h(u) \leqslant h(v) + 1.$

Operacja prześlij(u, v) może zostać wykonana, gdy spełnione są warunki:
(1) $e(u) > 0$
(2) $c_g(u, v) > 0$
(3) $h(u) = h(v) + 1$

prześlij(u, v)

$d_g(u, v) := \min\{e(u), c_g(u, v)\}$
if $(u, v) \in A$ **then**
    $g(u, v) := g(u, v) + d_g(u, v)$
else
    $g(u, v) := g(u, v) - d_g(u, v)$
$e(u) := e(u) - d_g(u, v)$
$e(v) := e(v) + d_g(u, v)$

Operacja podnieś(u) może zostać wykonana, gdy spełnione są warunki:
(1) $e(u) > 0$
(2) $\forall (u, v) \in A_g : \quad h(u) \leqslant h(v)$.

podnieś(u)

$h(u) := 1 + \min\{h(v) : \ (u, v) \in A_g\}$

inicjuj_przedprzepływ(G)

for każdy wierzchołek $v \in V$
    $h(v) := 0$
    $e(v) := 0$
for każdy łuk $(u, v) \in A$
    $g(u, v) := 0$
$h(s) := |V|$
for każdy łuk $(s, v) \in A$
    $g(s, v) := c(s, v)$
    $e(v) := c(s, v)$
    $e(s) := e(s) - c(s, v)$

Algorytm przedprzepływowy

inicjuj_przedprzepływ(G)
while można zastosować prześlij lub podnieś
    wybierz jedną z dopuszczalnych i wykonaj

$T = O(|V|^2|A|)$

Warianty zagadnienia przepływowego

1. Sieci z wieloma źródłami i ujściami.
2. Sieci z dolną przepustowością.
3. Przepływ rozszerzony.
4. Przepływ o minimalnym koszcie.
5. Sieci uogólnione.

Sieć z wieloma źródłami i ujściami

to sieć postaci $G_{ST} = (V, A, S, T, c)$, gdzie (V, A) jest grafem skierowanym, w którym S jest zbiorem źródeł, T jest zbiorem ujść, a c jest funkcją przepustowości

$$c : \; A \mapsto \mathbb{R}_{+}.$$

Przepływem w sieci G_{ST} nazywamy funkcję

$$f : \; A \mapsto \mathbb{R}_{+}$$

spełniającą warunki:
(1) $\forall (u, v) \in A : \quad 0 \leqslant f(u, v) \leqslant c(u, v)$
(2) $\forall v \in V \setminus (S \cup T) :$

$$\sum_{u \in V:\, (u,v) \in A} f(u, v) = \sum_{u \in V:\, (v,u) \in A} f(v, u).$$

Wartością przepływu f jest
$|f| = \sum_{(u,v)\in A,\, u\in S} f(u,v) = \sum_{(u,v)\in A,\, v\in T} f(u,v)$.

Problem maksymalnego przepływu w G_{ST}

Wejście: Sieć $G_{ST} = (V, A, S, T, c)$.
Wyjście: Przepływ f_{max} :
$|f_{max}| = \max\{|f| : \ f \text{ jest przepływem w } G_{ST}\}$.

Siecią pomocniczą dla $G_{ST} = (V, A, S, T, c)$ jest sieć
$G' = (V', A', s', t', c')$, w której:
$V' = V \cup \{s', t'\}$
$A' = A \cup \{(s', s) : \text{dla każdego } s \in S\} \cup \{(t, t') : \text{dla każdego } t \in T\}$

$$c'(u,v) = \begin{cases} c(u,v) & \text{jeśli } (u,v) \in A \\ \infty & \text{wpp} \end{cases}$$

Sieć z dolną przepustowością

to sieć postaci $G_b = (V, A, s, t, b, c)$, gdzie (V, A) jest grafem skierowanym, s jest źródłem, t ujściem, a b i c są odpowiednio funkcjami dolnej i górnej przepustowości

$$b, c : \; A \mapsto \mathbb{R}_+ .$$

Przepływem (*dopuszczalnym*) w sieci G_b nazywamy funkcję

$$f : \; A \mapsto \mathbb{R}_+$$

spełniającą warunki:
(1) $\forall (u, v) \in A : \quad b(u, v) \leqslant f(u, v) \leqslant c(u, v)$
(2) $\forall v \in V \setminus \{s, t\} :$

$$\sum_{u \in V:\, (u,v) \in A} f(u, v) = \sum_{u \in V:\, (v,u) \in A} f(v, u).$$

Problem maksymalnego przepływu w G_b

Wejście: Sieć $G_b = (V, A, s, t, b, c)$.
Wyjście: Przepływ f_{max} :
$|f_{max}| = \max\{|f| : f$ jest przepływem w $G_b\}$, lub NULL jeśli f nie istnieje.

Warunki konieczne na istnienie przepływu dopuszczalnego

$$\forall v \in V \setminus \{s, t\} : \sum_{u \in V} b(u, v) \leqslant \sum_{u \in V} c(v, u)$$

$$\forall v \in V \setminus \{s, t\} : \sum_{u \in V} b(v, u) \leqslant \sum_{u \in V} c(u, v)$$

Siecią pomocniczą dla sieci G_b jest sieć $G' = (V', A', s', t', c')$, w której:

$$V' = V \cup \{s', t'\}$$

$$A' = A \cup \{(s', v) : \text{dla każdego } v \in V \setminus s\}$$
$$\cup\{(v, t') : \text{dla każdego } v \in V \setminus t\} \cup (t, s)$$

$$c'(u, v) = \begin{cases} \sum_{w \in V} b(w, v) & \text{jeśli } u = s' \\ \sum_{w \in V} b(u, w) & \text{jeśli } v = t' \\ c(u, v) - b(u, v) & \text{jeśli } (u, v) \in A \\ \infty & \text{dla } (t, s) \end{cases}$$

Twierdzenie

Sieć G_b posiada przepływ dopuszczalny wtedy i tylko wtedy, gdy maksymalny przepływ w sieci pomocniczej G' jest równy $|f'| = \sum_{v \in V} c'(s', v) = \sum_{v \in V} c'(v, t')$.

Maksymalny przepływ w G_b

if znajdź_przepływ_dopuszczalny()==0 **then**
    return NULL
for każdy łuk $(u, v) \in A$ **do**
    $f(u, v) := f'(u, v) + b(u, v)$
    $c_f(u, v) := c(u, v) - f(u, v)$
    $c_f(v, u) := f(u, v) - b(u, v)$
while istnieje ścieżka powiększająca p w G_f **do**
    $c_f(p) := \min\{c_f(u, v) : (u, v) \in p\}$
    for każdy łuk $(u, v) \in p$ **do**
        $f(u, v) := f(u, v) + c_f(p)$
        $c_f(u, v) := c(u, v) - f(u, v)$
        $c_f(v, u) := f(u, v) - b(u, v)$

Przepływem (*rozszerzonym*) w sieci G_b nazywamy funkcję

$$f : A \mapsto \mathbb{R}_+$$

spełniającą warunki:
(1) $\forall (u,v) \in A : \quad b(u,v) \leqslant f(u,v) \leqslant c(u,v)$ lub $f(u,v) = 0$
(2) $\forall v \in V \setminus \{s,t\}$:

$$\sum_{u \in V:\ (u,v) \in A} f(u,v) = \sum_{u \in V:\ (v,u) \in A} f(v,u).$$

Problem maksymalnego przepływu rozszerzonego w G_b

Wejście: Sieć $G_b = (V, A, s, t, b, c)$.
Wyjście: Przepływ f_{max} :
$|f_{max}| = \max\{|f| : \ f$ jest przepływem rozszerzonym w $G_b\}$, lub NULL jeśli f nie istnieje.

Kosztem w sieci $G = (V, A, s, t, c)$ nazywamy funkcję $k : A \mapsto \mathbb{R}^+$.

Niech f będzie przepływem w sieci G. *Kosztem przepływu f* jest wartość

$$k_f = \sum_{(u,v) \in A} f(u, v) k(u, v).$$

Problem przepływu o minimalnym koszcie w G

Wejście: Sieć $G = (V, A, s, t, c)$ wraz z funkcją kosztu k, oraz $d \in R_+$.
Wyjście: Przepływ $f_{mink} : k_{f_{mink}} = \min\{k_f : f$ jest przepływem w G i $|f| = d\}$.

Twierdzenie

Przepływ f w G jest przepływem o minimalnym koszcie wtedy i tylko wtedy, gdy w sieci residualnej G_f nie istnieje cykl o ujemnym koszcie.

Przepływ o minimalnym koszcie w G

f := znajdź_przepływ(d)
G_f := sieć residualna dla f w G
while istnieje w G_f cykl C o koszcie ujemnym **do**
    modyfikuj f wzdłuż cyklu C
    modyfikuj G_f

Sieć uogólniona

to sieć postaci $G_g = (V, A, s, t, c, \gamma)$, gdzie (V, A) jest grafem skierowanym, s jest źródłem, t ujściem, c funkcją przepustowości, natomiast γ jest *funkcją zysku* (straty)
$\gamma : \ A \mapsto \mathbb{R}^+$.

Przepływem uogólnionym w sieci G_g nazywamy funkcję
$f : \ A \mapsto \mathbb{R}_+$ spełniającą warunki:
(1) $\forall (u, v) \in A : \ \ 0 \leqslant f(u, v) \leqslant c(u, v)$
(2) $\forall v \in V \setminus \{s, t\}$:

$$\sum_{u \in V\colon (v,u) \in A} f(v, u) = \sum_{u \in V\colon (u,v) \in A} f(u, v)\gamma(u, v).$$

Wartością przepływu f jest liczba
$|f| = \sum_{(u,t) \in A} f(u, t)\gamma(u, t)$.

Problem przepływu maksymalnego w sieci G_g

Wejście: Sieć $G_g = (V, A, s, t, c, \gamma)$.
Wyjście: Przepływ f_{max} :
$|f_{max}| = \max\{|f| : f$ jest przepływem w $G_g\}$.

Cykl C nazywamy *cyklem generującym przepływ* jeśli $\gamma(C) = \prod_{(u,v)\in C} \gamma(u, v) > 1$. Jeśli $\gamma(C) < 1$ to wówczas mówimy o *cyklu absorbującym przepływ*. Gdy $\gamma(C) = 1$ to cykl jest *jednostkowy*.

Uogólnioną ścieżką powiększającą (GAP) nazywamy cykl C generujący przepływ, wraz z dołączoną (dopuszczalnie pustą) ścieżką P od jednego z wierzchołków cyklu C do ujścia t w sieci residualnej G_f.

Twierdzenie

Przepływ f jest przepływem maksymalnym w G_g wtedy i tylko wtedy, gdy w sieci residualnej G_f nie istnieje uogólniona ścieżka powiększająca.

Przepływ maksymalny w G_g

$f := 0$
$G_f := G_g$
while istnieje w G_f uogólniona ścieżka powiększająca CP **do**
    modyfikuj f wzdłuż CP
    modyfikuj G_f

Problem minimalnego przekroju w grafie

Wejście: Graf $G = (V, E)$ oraz funkcja wagowa $w: \; E \mapsto \mathbb{R}_+$
Wyjście: Przekrój (X, Y):

$$w(X, Y) = \min\{w(X', Y') : \; (X', Y') \text{ jest przekrojem w } G\}.$$

$w(X, Y) = \sum_{\{x,y\} \in E:\, x \in X,\, y \in Y} w(\{x, y\})$

Niech $A \subset V$. Jeśli $v \notin A$ to $w(A, v) = \sum_{\{a,v\} \in E:\, a \in A} w(\{a, v\})$.
Wierzchołek $z \notin A$ nazywamy *najsilniej połączonym z A* jeśli $w(A, z) = \max\{w(A, v) : \; v \notin A\}$.

redukuj(G, v_0)

$A := \{v_0\}$
while $A \neq V$ **do**
    dołącz do A wierzchołek najsilniej połączony
$(X, Y) := (t, V \setminus \{t\})$
złącz w G dwa ostatnio dołączone do A wierzchołki s i t
return (X, Y)

Algorytm Stoera-Wagnera

$w_{min} := w(v_0, V \setminus \{v_0\})$
while $|V| > 1$ **do**
    (X, Y)=redukuj(G, v_0)
    if $w(X, Y) < w_{min}$ **then**
        $(X, Y)_{min} := (X, Y)$
        $w_{min} := w(X, Y)$
return $(X, Y)_{min}$, w_{min}

$T = \Theta(|V||E| + |V|^2 \log |V|)$.

Lemat

Niech s i t będą dwoma wierzchołkami w grafie $G = (V, E)$. Niech $G/\{s, t\}$ oznacza graf otrzymany z G poprzez złączenie wierzchołków s i t. Wówczas minimalnym przekrojem w G jest albo przekrój (S, T) albo minimalny przekrój w grafie $G/\{s, t\}$.

Lemat

Przekrój zwracany przez funkcję redukuj() jest minimalnym (S, T)-przekrojem zadanego na wejściu grafu G, gdzie s i t są ostatnimi dołączonymi wierzchołkami.

Skojarzeniem w grafie $G = (V, E)$ nazywamy podzbiór $M \subset E$ wierzchołkowo rozłącznych krawędzi.

Licznością skojarzenia M jest liczba $|M|$.

Mówimy, że M jest *najliczniejszym* skojarzeniem w G jeśli $|M| = \max\{|M'| : M'$ jest skojarzeniem w $G\}$. Wówczas liczbę $\mu(G) = |M|$ nazywamy *liczbą skojarzeniową* grafu G.

Problem najliczniejszego skojarzenia w grafie

Wejście: Graf $G = (V, E)$.
Wyjście: Najliczniejsze skojarzenie M w G.

Skojarzenie M w grafie $G = (V, E)$ nazywamy *pełnym* jeśli $|M| = |V|/2$.

Twierdzenie Tutte'a

Graf G posiada pełne skojarzenie wtedy i tylko wtedy, gdy $|S| \geqslant c$ dla każdego podzbioru $S \subset V$, gdzie c oznacza liczbę spójnych składowych o nieparzystym rzędzie w grafie $G[V \setminus S]$.

Twierdzenie Halla'a

Graf dwudzielny $G(X, Y; E)$ zawiera skojarzenie rozmiaru $|X|$ wtedy i tylko wtedy, gdy dla każdego podzbioru $S \subset X$ zachodzi $|S| \leqslant |N(S)|$.

Siecią pomocniczą dla grafu dwudzielnego $G = (X, Y; E)$ jest sieć $G = (V, A, s, t, c)$, w której:
$V = X \cup Y \cup \{s, t\}$
$A = \{(x, y) : \text{dla każdego } \{x, y\} \in E\} \cup \{(s, x) : \text{dla każdego } x \in X\} \cup \{(y, t) : \text{dla każdego } Y \in Y\}$
$c(u, v) = 1$ dla każdego łuku $(u, v) \in A$.

Twierdzenie

Graf dwudzielny $G = (X, Y; E)$ zawiera skojarzenie o liczności $|M|$ jeżeli istnieje przepływ f o wartości $|f| = |M|$ w sieci pomocniczej $G = (V, A, s, t, c)$.

Ścieżka $P =< v_0, v_1, \ldots, v_k >$ o nieparzystej długości k jest *ścieżką powiększającą* względem skojarzenia M w grafie G jeśli $\forall l: \ 1 \leqslant l \leqslant \frac{k-1}{2} : \{v_{2l-1}, v_{2l}\} \in M$, a wierzchołki v_0 oraz v_k są *wolne* (czyli nie są incydentne z żadną krawędzią skojarzenia M).

Twierdzenie Berge'a

M jest najliczniejszym skojarzeniem w grafie G wtedy i tylko wtedy, gdy G nie zawiera ścieżki powiększającej względem M.

Kielich to cykl o nieparzystej długości $2k+1$, w którym k krawędzi należy do skojarzenia M.
Podstawą kielicha jest albo wierzchołek wolny, albo skojarzony z krawędzią z M nie należącą do kielicha.

Zwinięciem kielicha B do podstawy y nazywamy taką transformację grafu $G=(V,E)$ do grafu $G'=(V',E')$, że
$V' = V \setminus (V(B) \setminus \{y\})$, oraz
$E' = E \setminus \{\{u,v\}:\ u \in V(B) \text{ lub } v \in V(B)\} \cup$
$\{\{y,v\}:\ \{u,v\} \in E,\ u \in V(B),\ v \notin V(B)\}$.
Operację odwrotną nazywamy *rozwinięciem kielicha B*.

Niech S oznacza zbiór wierzchołków wolnych względem skojarzenia M. *Lasem naprzemiennym* nazywamy las F, w którym:
- każdy wierzchołek ze zbioru S jest korzeniem w F
- każda krawędź o nieparzystej odległości od korzenia należy do M.

Każdy wierzchołek w F mający nieparzystą odległość od korzenia ma stopień 2 i nazywany jest wierzchołkiem *wewnętrznym*. Pozostałe wierzchołki są *zewnętrzne*.

Algorytm Edmondsa

$M = \emptyset$
$F = V(G)$
while (1) **do**
    if istnieje zewnętrzny wierzchołek $x \in F$ sąsiedni z $y \notin F$ **then**
        znajdź wierzchołek z: $\{y, z\} \in M$
        $F \cup \{x, y\} \cup \{y, z\}$
    else
    if istnieją zewnętrzne wierzchołki $x_1, x_2 \in F : \{x_1, x_2\} \in E$ **then**
        if x_1 oraz x_2 należą do różnych składowych w F **then**
            $p_1 :=$ ścieżka od $root(x_1)$ do x_1 w F
            $p_2 :=$ ścieżka od $root(x_2)$ do x_2 w F
            modyfikuj M wzdłuż ścieżki powiększającej $p_1 \cup \{x_1, x_2\} \cup p_2$
            $F :=$ zbiór wierzchołków wolnych względem M
        else
            $B :=$ kielich zamknięty przez $\{x_1, x_2\}$ o podstawie y
            zwiń B do y
            modyfikuj M względem B
    else
        break
for każdy zwinięty kielich B **do**
    rozwiń B
    modyfikuj M względem B

$T = O(|V|^2|E|)$

Warianty zagadnienia

1. Najliczniejsze skojarzenie o minimalnej wadze.
2. Najliczniejsze skojarzenie o maksymalnej wadze.
3. Skojarzenie o maksymalnej wadze.

Przez *płaską reprezentację* grafu $G = (V, E)$ rozumiemy takie rozmieszczenie wierzchołków zbioru V na płaszczyźnie, że jedyny punkt przecięcia dowolnych dwóch krawędzi to ewentualnie ich wspólny koniec.

Graf $G = (V, E)$ nazywamy *planarnym* jeśli posiada płaską reprezentację.

Testowanie planarności grafu

Wejście: Graf $G = (V, E)$.
Wyjście: Graf jest/nie jest planarny.

Znajdowanie płaskiej reprezentacji

Wejście: Graf $G = (V, E)$.
Wyjście: Płaska reprezentacja grafu G - jeśli istnieje.

Rozdzieleniem krawędzi $\{u, v\}$ w grafie $G = (V, E)$ nazywamy dodanie nowego wierzchołka w oraz zastąpienie tej krawędzi przez dwie krawędzie $\{u, w\}$ i $\{w, v\}$.
Rozdzieleniem grafu G nazywamy graf G' powstały z G poprzez wykonanie kolejnych rozdzieleń krawędzi.

Twierdzenie Kuratowskiego

Graf G jest planarny wtedy i tylko wtedy, gdy nie zawiera podgrafu będącego rozdzieleniem grafu K_5 lub $K_{3,3}$.

Zwinięciem krawędzi $\{u, v\}$ w grafie $G = (V, E)$ nazywamy operację złączenienia wierzchołków u oraz v wraz z usunięciem powstałej pętli oraz zastąpienia potencjalnych równoległych krawędzi przez pojedynczą krawędź.
Minorem grafu G nazywamy graf G' powstały z G poprzez wykonanie kolejnych operacji usuwania wierzchołków, usuwania krawędzi oraz/lub zwijania krawędzi.

Twierdzenie Wagnera

Graf G jest planarny wtedy i tylko wtedy, gdy nie posiada minorów będących K_5 lub $K_{3,3}$.

Dla danego grafu $G = (V, E)$ oraz jego podgrafu $G' = (V', E')$, poprzez *fragment* rozumiemy każdy podgraf grafu G, który jest albo:
(1) taką pojedynczą krawędzią $\{u, v\}$, że $\{u, v\} \notin E'$ ale $u, v \in V'$, albo
(2) podgrafem indukowanym przez spójną składową C w $G \setminus G'$, wraz ze wszystkimi krawędziami $\{u, v\}$ postaci $u \in V(C)$, $v \in V'$.
Wierzchołkami *zewnętrznymi* fragmentu nazywamy wierzchołki należące do V'.

α-ścieżka to dowolna ścieżka zawarta we fragmencie, której wierzchołki końcowe są wierzchołkami zewnętrznymi.

Algorytm Demoucron, Malgrange, Pertuiset

G' := dowolny cykl w G
while (1) **do**
    wyznacz wszystkie ściany w G'
    wyznacz zbiór P wszystkich fragmentów w G ze względu na G'
    if $P == \emptyset$ **then**
        G jest planarny
        G' jest płaską reprezentacją grafu G
        return
    for każdy fragment $p \in P$ **do**
        wyznacz $F(p)$
    if istnieje fragment p: $f(p) == 0$ **then**
        G nie jest planarny
        return
    if istnieje fragment p: $f(p) == 1$ **then**
        $r := p$
    else
        r := dowolny inny fragment
    wyznacz dowolną α-ścieżkę w r
    umieść α w ścianie $f \in F(r)$
    $G' := G' \cup \alpha$

$T = O(|V|^2)$

Niech G będzie grafem 2-spójnym.

Algorytm Hopcrofta-Tarjana

if $|E| > 3|V| - 6$ **then**
    return "nie jest planarny"
for każdy wierzchołek v w G **do**
    d[v] := 0
    $\pi[v] := \emptyset$
lv := 0
DFS_etykietowanie(v_0)
konwertuj(G, G^*)
sortuj_listy_sąsiadów()
DFS_testowanie(v_0)

$T = O(|V|)$

Oznaczenia

$d[v]$ - etykieta wierzchołka v w porządku DFS
$\pi[v]$ - poprzednik v w DFS
$P_T(v)$ oznacza zbiór potomków wierzchołka v w drzewie T, wraz z v
$S_T(v) = \{d[u] : \{w, u\} \in E$ oraz $d[w] > d[u]$ dla pewnego $w \in P_T(v)\}$
$L1(v) = \min\{S_T(v)\}$
$L2(v) = \min\{S_T(v) \setminus \{L1(v)\}\}$

DFS_etykietowanie(v)

```
lv++
d[v] := lv
L1[v] := lv
L2[v] := lv
for każdy sąsiad u wierzchołka v w G do
    if d[u] == 0 then
        π[u] := v
        DFS_etykietowanie(u)
    else
        if d[u] < L1[v] then
            L2[v] := L1[v]
            L1[v] := d[u]
        else
        if d[u] > L1[v] then
            L2[v] := min{d[u], L2[v]}
```

DFS_etykietowanie(v) (cd.)

if d[v] > 1 **then**
    if L1[v] $<$ L1[π[v]] **then**
        L2[π[v]] := min$\{L2[v], L1[\pi[v]]\}$
        L1[π[v]] := L1[v]
    else
    if L1[v] == L1[π[v]] **then**
        L2[π[v]] := min$\{L2[v], L2[\pi[v]]\}$
    else
        L2[π[v]] := min$\{L1[v], L2[\pi[v]]\}$

Graf (V, E) zastąpiony zostanie grafem zorientowanym $G^* = (V, A)$:
Niech $\{u, v\} \in E$: $d[u] < d[v]$.
$(u, v) \in A$ jeśli $\{u, v\}$ jest krawędzią drzewa T,
w przeciwnym przypadku $(v, u) \in A$.

Etykietowanie łuków

$$w(u,v) = \begin{cases} 2d[v] & \text{jeśli } (u,v) \text{ jest łukiem zwrotnym} \\ 2L1(v) & \text{jeśli } (u,v) \text{ jest łukiem drzewa oraz } L2(v) \geqslant d[u] \\ 2L1(v) + 1 & \text{jeśli } (u,v) \text{ jest łukiem drzewa oraz } L2(v) < d[u] \end{cases}$$

DFS_testowanie(v)

```
for każdy łuk (v, u) ∈ A do
    if d[v] < d[u] then
        w := L1[u]
        if ścieżka v → w nie koliduje z L_i then
            umieść v → w w L_i
        else
        if ścieżka v → w nie koliduje z L_o then
            umieść v → w w L_o
        else
            m=zamień_strony(v → w)
            if m==0 then
                return "nie jest planarny"
            if m==1 then
                umieść v → w w L_i
            else
                umieść v → w w L_o
        DFS_testowanie(u)
    else
```

DFS_testowanie(v) (cd.)

```
if ścieżka v → u nie koliduje z L_i then
    umieść v → u w L_i
else
if ścieżka v → u nie koliduje z L_o then
    umieść v → u w L_o
else
    m=zamień_strony(v → u)
    if m==0 then
        return "nie jest planarny"
    if m==1 then
        umieść v → u w L_i
    else
        umieść v → u w L_o
```

zamień_strony($u \rightarrow v$)

while (1) **do**
    B := pobierz_blok_ze_stosu()
    B' := pobierz_blok_ze_stosu()
    if czołowa cięciwa z L_o znajduje się w B_o **then**
        if czołowa cięciwa z L_i znajduje się w B_i **then**
            return 0
        if czołowa cięciwa z L_i znajduje się w B_i' **then**
            zamień cięciwy pomiędzy B_i i B_o
            połóż_na_stos(złącz B z B')
            znajdź nowe czołowe cięciwy w L_i oraz L_o
            if $u \rightarrow v$ nie koliduje z L_o **then**
                return -1
        else
            połóż_na_stos(złącz B z B')
    else

zamień_strony($u \to v$) (cd.)

if czołowa cięciwa z L_o znajduje się w B'_o **then**
    zamień cięciwy pomiędzy B_i i B_o
    połóż_na_stos(złącz B z B')
    znajdź nowe czołowe cięciwy w L_i oraz L_o
    if $u \to v$ nie koliduje z L_i **then**
        return 1
else
    połóż_na_stos(złącz B z B')

Zbiorem niezależnym w grafie $G = (V, E)$ nazywamy taki podzbiór $I \subset V$, że żadne dwa wierzchołki w I nie są sąsiednie.

Mówimy, że zbiór niezależny I w grafie G jest *najliczniejszy*, jeśli nie istnieje inny zbiór niezależny w G o rzędzie większym niż $|I|$. Wówczas $\alpha(G) = |I|$ nazywamy *liczbą niezależności* grafu G.

Najlicznieszy zbiór niezależny

Wejście: Graf $G = (V, E)$
Wyjście: Najliczniejszy zbiór niezależny I w G

Najliczniejszy zbiór niezależny jest problemem wielomianowym dla wielu klas grafów, m.in.:
- dwudzielnych
- doskonałych (w tym przekątniowych oraz przedziałowych)

Kliką w grafie $G = (V, E)$ nazywamy taki podzbiór $Q \subset V$, że każde dwa wierzchołki w Q są sąsiednie.

Mówimy, że klika Q w grafie G jest *najliczniejsza*, jeśli nie istnieje klika w G rzędu większego niż $|Q|$.
Wówczas $\omega(G) = |Q|$ nazywamy *liczbą klikową* grafu G.

Twierdzenie

Zbiór Q jest kliką w grafie $G = (V, E)$ wtedy i tylko wtedy, gdy Q jest zbiorem niezależnym w dopełnieniu $\overline{G} = (V, \overline{E})$ grafu G.
Zachodzi zatem zależność $\omega(G) = \alpha(\overline{G})$.

Najliczniejsza klika

Wejście: Graf $G = (V, E)$
Wyjście: Najliczniejsza klika Q w G

Najliczniejsza klika jest problemem wielomianowym dla wielu klas grafów, m.in.:
- o ograniczonym przez stałą stopniu maksymalnym
- planarnych
- doskonałych (w tym przekątniowych oraz przedziałowych)
- będących dopełnieniami grafów dwudzielnych

Definicja

Kolorowaniem (wierzchołkowym) grafu $G = (V, E)$ nazywamy odwzorowanie $c: V \mapsto C$, gdzie C oznacza zbiór kolorów.
Jeśli dla każdych dwóch sąsiednich wierzchołków v, u zachodzi $c(v) \neq c(u)$, to kolorowanie nazywamy *właściwym*.

Jeśli $|C| = k$ to mówimy o *k-kolorowaniu*.
Minimalną liczbę k, dla której istnieje k-kolorowanie właściwe wierzchołków grafu G nazywamy *liczbą chromatyczną* grafu G i oznaczamy $\chi(G)$.

Zbiór wierzchołków pokolorowanych tym samym kolorem nazywamy *klasą kolorową*.
Każda klasa kolorowa w kolorowaniu właściwym jest zbiorem niezależnym.

Kolorowanie wierzchołkowe grafu

Wejście: Graf $G = (V, E)$ oraz stała $k \in \mathbb{N}$
Wyjście: Graf jest/nie jest wierzchołkowo właściwie k-kolorowalny.

Wyznaczenie liczby chromatycznej

Wejście: Graf $G = (V, E)$
Wyjście: $\chi(G)$

Lemat

Dla każdego grafu G zachodzi $1 \leqslant \chi(G) \leqslant n$.

Lemat

Niech G będzie grafem, w którym $\Delta(G) \geqslant 1$. Wówczas $\chi(G) = 2$ wtedy i tylko wtedy gdy G jest grafem dwudzielnym.

Lemat

W każdym grafie G zachodzą zależności:

$$\chi(G) \geqslant \omega(G)$$

$$\chi(G) \geqslant \lceil \frac{n}{\alpha(G)} \rceil$$

Twierdzenie

Dla każdego grafu G zachodzi

$$\chi(G) \begin{cases} = & \Delta(G)+1 \quad \text{gdy } G = K_n \text{ lub } G = C_{2p+1} \\ \leqslant & \Delta(G) \quad \text{wpp} \end{cases}$$

Twierdzenie

Jeśli G jest grafem planarnym, to $\chi(G) \leqslant 4$.

Twierdzenie

Jeśli G jest grafem planarnym nie zawierającym trójkątów, to $\chi(G) \leqslant 3$.

Kolorowanie wierzchołkowe jest problemem wielomianowym dla wielu klas grafów, m.in.:
- podkubicznych
- planarnych bez trójkątów
- doskonałych (w tym przekątniowych oraz przedziałowych)

Definicja

Krawędziowym k-kolorowaniem (właściwym) grafu $G = (V, E)$ nazywamy takie odwzorowanie $c : E \mapsto C$, gdzie C oznacza zbiór kolorów oraz $|C| = k$, że dla każdych dwóch incydentnych krawędzi e_1, e_2 zachodzi $c(e_1) \neq c(e_2)$.

Minimalną liczbę k, dla której istnieje k-kolorowanie krawędzi grafu G nazywamy *indeksem chromatycznym* grafu G i oznaczamy $\chi'(G)$.

Zbiór krawędzi pokolorowanych tym samym kolorem nazywamy *klasą kolorową*.
Każda klasa kolorowa jest skojarzeniem.

Krawędziowe kolorowanie grafu

Wejście: Graf $G = (V, E)$ oraz stała $k \in \mathbb{N}$.
Wyjście: G jest/nie jest k-kolorowalny krawędziowo.

Wyznaczenie indeksu chromatycznego

Wejście: Graf $G = (V, E)$
Wyjście: $\chi'(G)$

Lemat

Dla każdego grafu G spełniona jest zależność $\chi'(G) \leqslant 2\Delta(G) - 1$.

Twierdzenie

Jeśli G jest grafem dwudzielnym, to $\chi'(G) = \Delta(G)$.

Twierdzenie

W dowolnym grafie G zachodzi $\chi'(G) \leqslant \lfloor \frac{3}{2}\Delta(G) \rfloor$.

Twierdzenie

Dla każdego grafu G zachodzi $\Delta(G) \leqslant \chi'(G) \leqslant \Delta(G) + 1$.

Wachlarzem o środku w wierzchołku v nazywamy ciąg wierzchołków $< u_1, u_2, \ldots, u_k >_v$ spełniający własności:
(1) $\{u_1, u_2, \ldots, u_k\}$ jest niepustym zbiorem parami różnych sąsiadów wierzchołka v,
(2) krawędź $\{v, u_1\}$ jest niepokolorowana, natomiast każda z krawędzi $\{v, u_i\}$ jest pokolorowana, $2 \leqslant i \leqslant k$,
(3) kolor krawędzi $\{v, u_i\}$ jest brakującym kolorem w palecie wierzchołka u_{i-1}, dla każdego $2 \leqslant i \leqslant k$.

Wachlarz, który nie może zostać powiększony poprzez dodanie kolejnej krawędzi incydentnej do v nazywamy *maksymalnym*.

znajdź_wachlarz_maksymalny(v, u_1)

a := brakujący kolor w palecie wierzchołka u_1
i := 2
while istnieje krawędź $\{v, u\}$ pokolorowana kolorem a oraz
    u nie należy do wachlarza **do**
  umieść $u_i := u$ w wachlarzu
  a := brakujący kolor w palecie wierzchołka u_i
  i++

odwróć_wachlarz($< u_1, u_2, \ldots, u_k >_v$)

for $i := 1$ **to** $k - 1$ **do**
  kolor[$\{v, u_i\}$] := kolor[$\{v, u_{i+1}\}$]
usuń kolor krawędzi $\{v, u_k\}$

Algorytm Misra-Gries

while istnieje niepokolorowana krawędź $\{v, u\}$ w G **do**
    if istnieje kolor a brakujący w paletach v i u **then**
        kolor$[\{v, u\}] := a$
    else
        znajdź_wachlarz_maksymalny(v, u)
        wyznacz wierzchołek brzegowy w wachlarza
        zamień kolory $a \leftrightarrow b$ wzdłuż ścieżki od v do w
        odwróć_wachlarz$(< u_1, u_2, \ldots, w >)$
        kolor$[\{v, w\}] := b$

$T = O(|V||E|)$

Graf $G = (V, E)$ jest grafem przekątniowym jeśli każdy cykl o długości co najmniej cztery posiada przekątną (krawędź łączącą dwa niekolejne wierzchołki w tym cyklu).

Rozpoznanie grafu przekątniowego

Wejście: Graf $G = (V, E)$
Wyjście: G jest/nie jest grafem przekątniowym

Porządek doskonale eliminujący (PEO) w grafie $G = (V, E)$ to taki porządek $v_1, v_2, \ldots, v_n$ wierzchołków, że zbiór $N_{nast}(v_i)$ tworzy klikę, dla każdego $i = 1, 2, \ldots, n-1$, gdzie $N_{nast}(v_i)$ oznacza zbiór sąsiadów wierzchołka v_i występujących po v_i w tym porządku.

Twierdzenie Fulkersona-Grossa

Graf G jest grafem przekątniowym wtedy i tylko wtedy gdy G posiada PEO.

Leksykograficzny BFS

Wejście: Graf $G = (V, E)$ oraz $v \in V$
Wyjście: Porządek Γ wierzchołków z V

LexBFS(G, v)

$n := |V|$
$L := \{V\}$
for $i := 0$ **to** n-1 **do**
    $u :=$ usuń wierzchołek ze zbioru A_1 w L
    if $A_1 == \emptyset$ **then**
        usuń(A_1)
    $\Gamma[i] := u$
    for każdy zbiór $A_j \in L$ **do**
        $B_j := A_j \cap N(u)$
        if B_j != A_j && B_j != $\emptyset$ **then**
            wstaw B_j bezpośrednio przed A_j w L
            $A_j := A_j \setminus B_j$
return Γ

$T = O(|V| + |E|)$

Niech $\pi(v)$ oznacza sąsiada v ostatniego przed v w porządku Γ

graf_przekątniowy(G, Γ)

for każdy wierzchołek v_i w Γ **do**
    if $N_{pop}(v_i) \not\subseteq N_{pop}(\pi(v_i)) \cup \{\pi(v_i)\}$ **then**
        return G nie jest grafem przekątniowym
return G jest grafem przekątniowym

$T = O(|V| + |E|)$

Lemat

Niech $< v_1, v_2, \ldots, v_n >$ będzie ciągiem wierzchołków grafu G w PEO. Wówczas algorytm zachłanny, odwiedzający wierzchołki w porządku $v_n, v_{n-1}, \ldots, v_1$ znajduje najliczniejszą klikę.

Lemat

Niech $< v_1, v_2, \ldots, v_n >$ będzie ciągiem wierzchołków grafu G w PEO. Wówczas algorytm zachłanny, odwiedzający wierzchołki w porządku $v_1, v_2, \ldots, v_n$ znajduje najliczniejszy zbiór niezależny.

Graf $G = (V, E)$ jest grafem przedziałowym jeśli reprezentuje graf przecięć przedziałów domkniętych na rzeczywistej osi liczbowej. Każdy wierzchołek w V odpowiada jednemu przedziałowi, dwa wierzchołki są sąsiednie jeśli odpowiadające im przedziały mają niepuste przecięcie.

Rozpoznanie grafu przedziałowego

Wejście: Graf $G = (V, E)$
Wyjście: G jest/nie jest grafem przedziałowym

Twierdzenie

Graf $G = (V, E)$ jest grafem przedziałowym wtedy i tylko wtedy, gdy wszystkie maksymalne kliki w G mogą zostać uporządkowane w ciąg $< Q_1, Q_2, \ldots, Q_q >$ o takiej własności, że jeśli $v \in Q_i$ oraz $v \in Q_j$, to również $v \in Q_h$ dla każdego $i < h < j$.

drzewo_klikowe(G,Γ)

$T := \emptyset$
for każdy wierzchołek v_i w Γ **do**
    if $N_{pop}(v_i) \mathrel{!}= N_{pop}(\pi(v_i)) \cup \{\pi(v_i)\}$ **then**
        stwórz nową klikę $C := \{v_i\} \cup N_{pop}(v_i)$
        $c(v_i) := C$
        $\pi(C) := c(\pi(v_i))$
        $T := T \cup \{C, \pi(C)\}$
    else
        $C(\pi(v_i)) := C(\pi(v_i)) \cup \{v_i\}$
        $c(v_i) := C(\pi(v_i))$
return T

graf_przedziałowy(G)

Γ := LexBFS(G, v_0)
if graf_przekątniowy(G, Γ) == FALSE **then**
    return FALSE
T := drzewo_klikowe(G, Γ)
uporządkowana lista $L := (V(T))$
$P := \emptyset$
while istnieje w L zbiór X nie będący singletonem **do**
    if $P == \emptyset$ **then**
        C := ostatnia klika w X względem indeksu w T
        zamień X na $X \setminus \mathcal{C}, \mathcal{C}$ w L
        $\mathcal{C} := \{C\}$
    else
        p := usuń dowolny element z P
        $\mathcal{C}$:= zbiór maksymalnych klik zawierających p
        X_a := pierwszy zbiór w L zawierający element z $\mathcal{C}$
        X_b := ostatni zbiór w L zawierający element z $\mathcal{C}$
        zamień X_a na $X_a \setminus \mathcal{C}, X_a \cap \mathcal{C}$ w L
        zamień X_b na $X_b \cap \mathcal{C}, X_b \setminus \mathcal{C}$ w L
    for każda pozostała w T krawędź (C_i, C_j): $C_i \in \mathcal{C}$ oraz $C_j \notin \mathcal{C}$
        $P := P \cup (C_i \cap C_j)$
        usuń $\{C_i, C_j\}$ z T

graf_przedziałowy(G) (cd.)

for każdy wierzchołek $v \in V(G)$
    if kliki zawierające v nie występują kolejno w L **then**
        return FALSE
return L

$$T = O(|V| + |E|)$$