Magdalena Stachowicz

28. **Praktyczne zastosowania grafów.** (*Zaprezentować problem spoza matematyki, który da się przełożyć na język grafów. Omówić zagadnienie teoretyczne modelujące podany problem praktyczny; podać algorytm (-y) służący(-e) do jego rozwiązania; omówić cechy tego algorytmu, takie jak poprawność i ewentualnie czas działania w zależności od rozmiaru grafu. Możliwe zagadnienia do wyboru:*

a) *Znajdowanie najkrótszej drogi od punktu A do punktu B, gdy dane są odległości do i między punktami pośrednimi. Algorytm Dijkstry -- lub inny--o złożoności wielomianowej. Ocena co najwyżej 4,0*

b) *Zagadnienie modelowane jako problem najkrótszych połączeń: wyznaczanie minimalnego drzewa spinającego w grafie spójnym z wagami. Algorytmy Prima i Kruskala, o złożoności wielomianowej. Ocena co najwyżej 4,5.*

c) *Zagadnienie modelowane jako problem komiwojażera: znajdowanie cyklu Hamiltona w grafie pełnym z wagami. Dokładny algorytm „siłowy" nie ma złożoności wielomianowej i czas jego wykonania dla dużych grafów jest nierealistyczny. Algorytmy przybliżone: najbliższego sąsiada i najtańszego łącza mają wielomianową złożoność, ale na ogół nie dają dokładnej odpowiedzi. Wyczerpująca odpowiedź może zasługiwać na 5.0)*

Grafem prostym nieskierowanym nazywamy układ $G = \langle V, E \rangle$ (V – wierzchołki, E – krawędzie)

V jest zbiorem niepustym a

$E \subseteq \{\{v,w\}: v, w \in V \wedge v \neq w\}$

Mówimy, że graf jest spójny gdy dowolne dwa jego wierzchołki można połączyć trasą.

Grafy eulerowskie - istnieje w nim ścieżka zamknięta przechodząca przez każdą krawędź dokładnie raz.

A) Zagadnienie najkrótszej drogi

Def. Grafem z wagami nazywamy układ: $G = \langle V, \mathcal{E}, \varphi \rangle$ taki, że $V \neq \emptyset$

$\langle V, \mathcal{E} \rangle$ - jest grafem prostym, nieskierowanym

$\varphi: \mathcal{E} \to \mathbb{R}_+ = (0, +\infty)$

Algorytm Dijkstry

Zmienne pomocnicze: v_i, $l()$, $t()$.

1. $v_1 := A$ („wystartuj z A")

$l(A) = (-, 0)$

Dla $i \neq 1$ połóż $t(v_i) = (-, \infty)$ (wszystkie inne wierzchołki uzyskują wagę nieskończoność)

2. Dopóki nie ustalono wartości $l(x)$ postępuj następująco:

a) Niech v będzie tym wierzchołkiem, który jako ostatni uzyskał

$l(v) = (l_1(v) \quad l_2(v)) = (l_1, l_2)$

Dla każdego wierzchołka sąsiadującego z v, który jeszcze nie ma $l(w)$, oblicz $l_2 + \varphi(vw)$

Niech $t(w)$ będzie (t_1, t_2)

Jeżeli $l_2 + \varphi(v, w) < t_2$ zmień wartość $t(w)$ na nowe $t(w) := (v, l_2 + \varphi(w))$

b) wśród wszystkich wierzchołków które nie mają jeszcze $l(w)$ wybierz ten dla którego liczba t_2 jest najmniejsza (gdy jest więcej niż jeden wybierz którykolwiek)

Zdefiniuj $l(w) := t(w)$

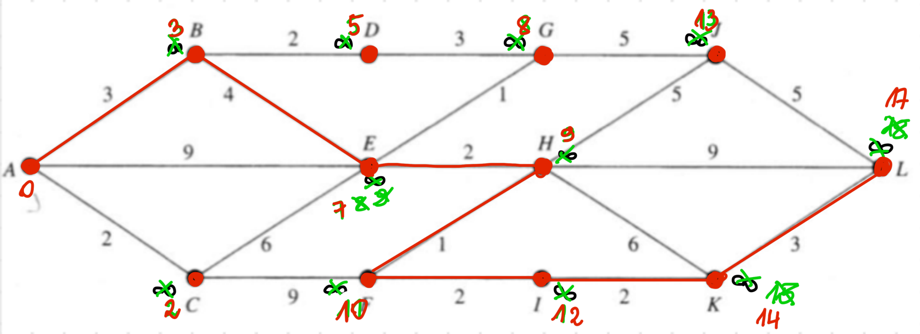
Terminologia $l(w)$ - etykieta stała

$t(w)$ - etykiety tymczasowe

Przykład.

Mamy daną mapę na której litery od A do L oznaczają miasta połączone drogami. Jaka jest długość najkrótszej drogi z miasta A do miasta L, jeśli długości dróg są takie jak zaznaczono na mapie?

I sposób



$l(A)=0$ $l(G)=8$

$l(B)=3$ $l(H)=9$

$l(C)=2$ $l(I)=12$

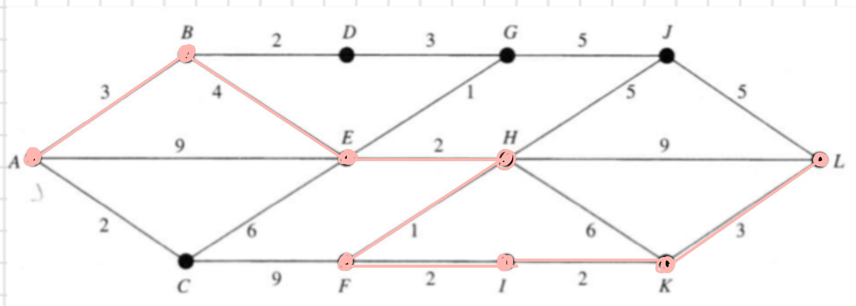
$l(D)=5$ $l(J)=13$

$l(E)=7$ $l(K)=14$

$l(F)=10$ $l(L)=17$

Najkrótsza droga z A do L ma długość 17.

II sposób



	A	B	C	D	E	F	G	H	I	J	K	L
(A,0)	0	(A,3)	(A,2)	∞	(A,9)	∞	∞	∞	∞	∞	∞	∞
(C,2)	0	(A,3)	(A,2)	∞	(A,8)	(A,11)	∞	∞	∞	∞	∞	∞
(B,3)	0	(A,3)	(A,2)	(A,5)	(A,7)	(A,11)	∞	∞	∞	∞	∞	∞
(D,5)	0	(A,3)	(A,2)	(A,5)	(A,7)	(A,11)	(A,8)	∞	∞	∞	∞	∞
(E,7)	0	(A,3)	(A,2)	(A,5)	(A,7)	(A,11)	(A,8)	(A,9)	∞	∞	∞	∞
(G,8)	0	(A,3)	(A,2)	(A,5)	(A,7)	(A,11)	(A,8)	(A,9)	∞	(A,13)	∞	∞
(H,9)	0	(A,3)	(A,2)	(A,5)	(A,7)	(A,10)	(A,8)	(A,9)	∞	(A,13)	(A,15)	(A,18)
(F,10)	0	(A,3)	(A,2)	(A,5)	(A,7)	(A,10)	(A,8)	(A,9)	(A,12)	(A,13)	(A,15)	(A,18)
(I,12)	0	(A,3)	(A,2)	(A,5)	(A,7)	(A,10)	(A,8)	(A,9)	(A,12)	(A,13)	(A,14)	(A,18)
(J,13)	0	(A,3)	(A,2)	(A,5)	(A,7)	(A,10)	(A,8)	(A,9)	(A,12)	(A,13)	(A,14)	(A,18)
(K,14)	0	(A,3)	(A,2)	(A,5)	(A,7)	(A,10)	(A,8)	(A,9)	(A,12)	(A,13)	(A,14)	(A,17)
L(17)	0	(A,3)	(A,2)	(A,5)	(A,7)	(A,10)	(A,8)	(A,9)	(A,12)	(A,13)	(A,14)	(A,17)

a) Złożoność obliczeniowa algorytmu Dijkstry

- maksymalna liczba pojedynczych kroków obliczeniowych, które algorytm musi wykonać, by zakończyć działanie, zależne od wielkości grafu.

milczące założenie: każdy pojedynczy krok zajmuje tyle samo czasu: jedna jednostka czasu.

złożoność obliczeniowa jest funkcją: zależy od wielkości egzemplarza, tzn. od wielkości grafu.

wielkość jest mierzona jako liczba wierzchołków grafu: n, lub liczba krawędzi: m.

jesteśmy zainteresowani „dobrym" ograniczeniem górnym tej liczby kroków.

w kroku startowym nadaję etykiety: "0", "∞".

w kroku i-tym $(i = 1, \dots n-1)$ gdy już i-wierzchołków uzyskało etykietę stałą mamy:

- „i" wierzchołków ze stałymi etykietami
- „$n-i$" wierzchołków ma etykiety tymczasowe

Dla każdego sąsiada wierzchołka, który ostatnio został zaetykietowany $l_2(v) + \varphi(v, \omega)$ - jedno dodawanie; porównuję tę liczbę z $l_2(\omega)$ - jedno porównanie i wybieram mniejszą.

Ostatnio zaetykietowany wierzchołek na stałe ma co najwyżej $n-i$ sąsiadów, spośród tych $n-i$ etykietek wybieram najmniejszą, wykonuję $n-i-1$ porównań.

W każdym spośród tych kroków: $1, \dots n-1$ wykonujemy $(n-i) \cdot 2 + (n-i-1)$ operacji podstawowych

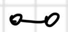
Sumarycznie wykonujemy:

$$\sum_{i=1}^{n-1} \left[(2(n-i) + n-i-1) \right] = \sum_{i=1}^{n-1} \left[\underbrace{(3n-1)}_{\text{stałe}} - 3i \right] = (3n-1) \cdot (n-1) - 3 \cdot \frac{(n-1) \cdot n}{2} = \frac{3}{2} n^2 \quad \frac{5}{2} n + 1$$

Złożoność obliczeniowa algorytmu Dijkstry to $\frac{3}{2} n^2 - \frac{5}{2} n + 1$; czyli f. kwadratowa

B) Wyznaczanie minimalnego drzewa spinającego w grafie spójnym z wagami.

Drzewem nazywamy graf spójny nie zawierający cykli. „Acykliczny" - nie zawierający cykli

Przykład. 

W grafie spójnym acyklicznym wierzchołki łączy jakaś droga.

Tw. Każde drzewo o n wierzchołkach ma $n-1$ krawędzi

Tw. Graf o n wierzchołkach jest drzewem wtw. gdy zachodzi jeden z poniższych warunków:

1) jest spójny i acykliczny;
2) każde dwa wierzchołki tego grafu połączone są dokładnie jedną drogą;
3) jest spójny, ale odejście dowolnej jednej krawędzi powoduje utratę spójności;
4) jest acykliczny, ale dodanie jednej krawędzi spowoduje powstanie cyklu;
5) jest spójny i ma $n-1$ krawędzi;
6) jest acykliczny i ma $n-1$ krawędzi.

Niech V_G - zbiór wierzchołków grafu G, E_G - zbiór krawędzi grafu G.
V_J - " - " - " - J, E_J - " - " - " - J

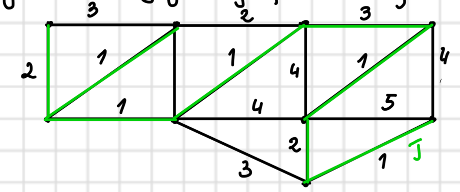
$G = \langle V_G, E_G \rangle$ $\quad J = \langle V_J, E_J \rangle$

Def. Mówimy, że graf J jest drzewem spinającym grafu G wtw gdy

1) J jest drzewem
2) $J \subseteq G$ $\quad (V_J \subseteq V_G,\ E_J \subseteq E_G)$
3) $V_J = V_G$

Wniosek: Drzewo spinające grafu o n wierzchołkach ma $n-1$ krawędzi.

Przykład. Dany jest graf G z wagami $\langle G, \varphi \rangle$. Znaleźć drzewo spinające o najmniejszej łącznej wadze wszystkich krawędzi.

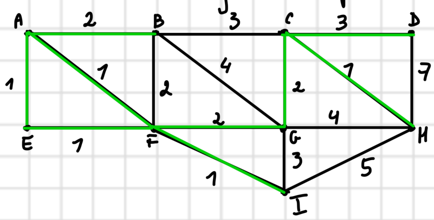


$\varphi(J) = 12$

Algorytm Prima

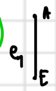
Krok podstawowy: wybór dowolnego wierzchołka v spośród krawędzi incydentnych z v, wybierz jedno e_1 - tę, która ma najmniejszą wagę. Niech $k=1$. Pętla dopóki $k < n-1$

Dla wybranych krawędzi e_1 do e_k oraz grafu J utworzonego z tych krawędzi, wybierz spośród krawędzi grafu G taką, która łączy wierzchołek v z J z jakimś x spoza J. Nazwij ją e_{k+1}. Ustanów nowe k o jeden większe od dotychczasowego.

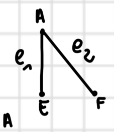


1) Wybierz dowolny wierzchołek (A)

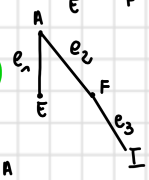
Wybieramy najtańszą krawędź 1. $AE = e_1$ (:1)
$k = 1$



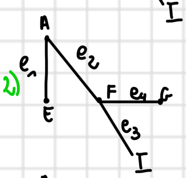
2. $AF = e_2$ (:1)
$k = 2$



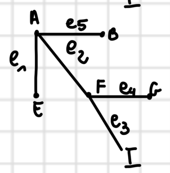
3. $FI = e_3$ (:1)
$k = 3$



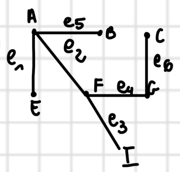
4. $FG = e_4$ (:2)
$k = 4$



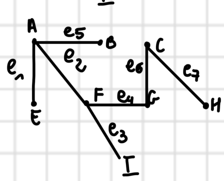
5. $AB = e_5$ (:2)
$k = 5$



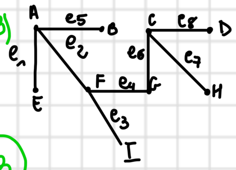
6. $CG = e_6$ (:2)
$k = 6$



7. $CH = e_7$ (:1)
$k = 7$



8. $CD = e_8$ (:3)
$k = 8$



13

Złożoność algorytmu Prima

Graf ma n wierzchołków

- w kroku 1; po wybraniu A; porównuje wagi wszystkich krawędzi wychodzących z A; maksymalnie jest ich $n-1$; porównań jest $n-2$; kroków obliczeniowych: $n-2$.
- $\left.\begin{matrix} A \\ B \end{matrix}\right\} n-2$ Badanych krawędzi jest co najwyżej $n-2$ z A, $n-2$ z B

$(n-2)+(n-2)+1$ porównań

- Gdy jest już wybranych k krawędzi $e_1, \dots e_k$

wierzchołków w J jest $k+1$

— " — " poza J jest $n-(k+1) = n-k-1$

dla każdego z $(k+1)$ wierzchołków mamy co najwyżej $n-k-1$ krawędzi do zbadania

$(k+1)(n-k-1)$ krawędzi do porównania

$(k+1)(n-k-1)-1$ porównań

$kn+n-k^2-k-k-1-1 = kn+n-k^2-2k-2 \rightarrow$ łączenie porównań wykonywanych w każdym kroku

$\sum_{k=1}^{n-2}(kn+n-k^2-2k-2) \rightarrow$ łączenie porównań wykonywanych sumarycznie

$$S = \sum_{k=1}^{n-2} kn + \sum_{k=1}^{n-2} n - \left(\sum_{k=1}^{n-2} k^2 + \sum_{k=1}^{n-2} 2k + \sum_{k=1}^{n-2} 2\right) = n \cdot \frac{(n-2)(n-1)}{2} + n(n-2) - \left[\frac{(n-2)(n-1)(2n-3)}{6} + 2 \cdot \frac{(n-2)(n-1)}{2} + 2(n-2)\right] = \frac{n^3-13n+18}{6}$$

- wielomian stopnia 3.

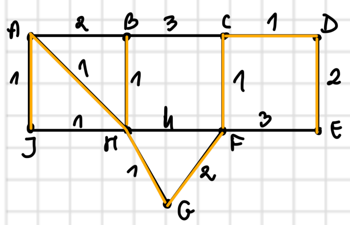
Rząd złożoności algorytmu Prima to n^3. Problem najkrótszych połączeń jest klasy P.

Algorytm Kruskala

Wypisujemy krawędzie w kolejności niemalejącej. Kolejno wybieramy krawędzie o najmniejszej wadze, ale tak by nie utworzyć cyklu aż wybierzemy $n-1$ krawędzi.

wierzchołki

Przykład.



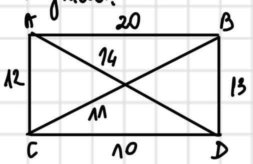
AH ✓
AJ ✓
~~JH~~ stworzy cykl z AH i AJ
HG ✓
CF ✓
BH ✓
CD ✓
~~AB~~ stworzy cykl z AH i BH
DE ✓
GF ✓ STOP
~~FE~~
~~BC~~
~~HF~~

C) Dany jest pełny graf z wagami o n wierzchołkach. Znaleźć cykl w tym grafie przechodzący przez wszystkie wierzchołki o najmniejszej łącznej wadze.

Def. W danym grafie G cykl przechodzący przez wszystkie wierzchołki nazywa się CYKLEM HAMILTONA.
Graf w którym istnieje cykl Hamiltona nazywa się hamiltonowski.

Problem komiwojażera - problem znalezienia minimalnego cyklu Hamiltona w grafie K_n.

Przykład.



1) ABCDA ≡ BCDAB ≡ ADCBA
2) ABDCA
3) ACBDA

ACDBA ≡ 2
ADBCA ≡ 3
ADCBA ≡ 1

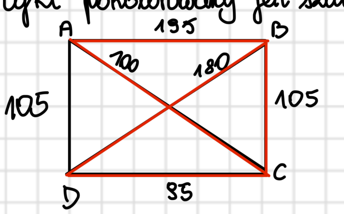
Liczba cykli do rozważania $\frac{3!}{2} = \frac{6}{2} = 3$

Ogólnie w grafie $K_n = \frac{(n-1)!}{2}$

ABCDA = 20 + 13 + 10 + 12 = 55
ABDCA = 20 + 11 + 10 + 14 = 55
ACBDA = 14 + 13 + 11 + 12 = 50 → najkrótsza

Algorytm najtańszego łącza

1. Wypisać krawędzie grafu w kolejności niemalejącej
2. Zaczynając od krawędzi o najmniejszej wadze, kolorujemy na czerwono kolejną o najmniejszej wadze spośród krawędzi niepokolorowanych, stosując się do następujących zasad:
   - nie mogą 3 czerwone krawędzie wychodzić z tego samego wierzchołka
   - nie wolno zamknąć czerwonego cyklu zanim cykl nie obejmie wszystkich wierzchołków grafu
3. Kończymy gdy ostatnia pokolorowana krawędź zamyka cykl - jest to cykl obejmujący wszystkie wierzchołki. Cykl pokolorowany jest szukanym rozwiązaniem przybliżonym.



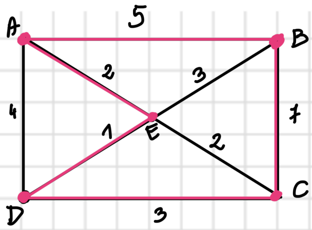
CD
AC
BC
AD
AB

ACDBA = 100 + 85 + 180 + 195 = 570

Algorytm najbliższego sąsiada

Działanie algorytmu

1. Wierzchołek początkowy oznaczamy jako odwiedzony i ustawiamy jako aktualny.
2. Znajdujemy najkrótszą spośród krawędzi łączących aktualny wierzchołek z jeszcze nieodwiedzonymi wierzchołkami.
3. Dołączamy do rozwiązania krawędź znalezioną w punkcie 2.
4. Wierzchołek będący *drugim końcem* krawędzi znalezionej w punkcie 2 oznaczamy jako odwiedzony i ustawiamy jako aktualny.
5. Jeśli są jeszcze nieodwiedzone wierzchołki, przechodzimy do punktu 2.
6. Dołączamy krawędź łączącą ostatnio dodany wierzchołek z wierzchołkiem początkowym. Zamykamy w ten sposób cykl.



AEDCBA

$2+1+3+7+5=18$