Mat Dys. Ćwiczenia nr 6
Drzewa rozpinające
Algorytm wyznaczania optymalnych drzew rozpinających
Kodowanie drzew

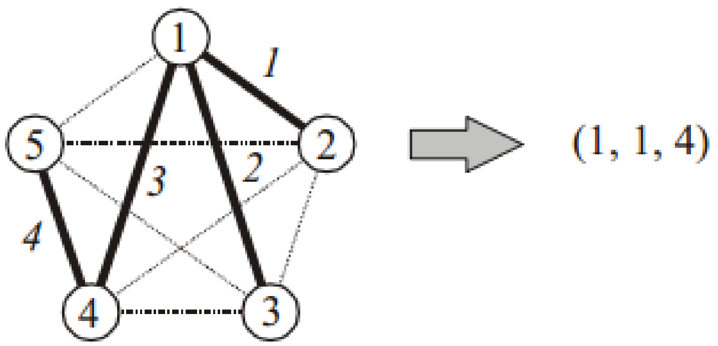
Algorytm wyznaczania kodu Prufera:

Aby wyznaczyć kod Prufera dla danego drzewa T na zbiorze wierzchołków $\{1,...,n\}$, należy:

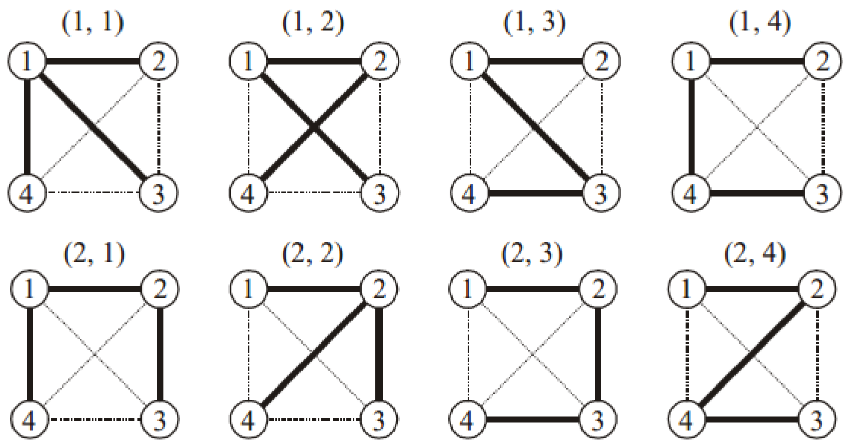
a) znaleźć najmniejszy wierzchołek stopnia jeden, powiedzmy v. Niech w będzie wierzchołkiem połączonym z v;
b) zapisać w oraz usunąć wierzchołek v wraz z krawędzią vw;
c) Jeśli w drzewie pozostała więcej niż jedna krawędź, to przejść do kroku **a)**; w przeciwnym razie zakończyć algorytm.

Otrzymany ciąg liczb jest kodem Prufera dla drzewa *T*.

Przykład kodowania drzewa: **kod Prufera <1, 1, 4>. Kolejne** Cyfry przy krawędziach oznaczają kolejność usuwania krawędzi.



Przykłady kodowania drzew w grafie pełnym K_4

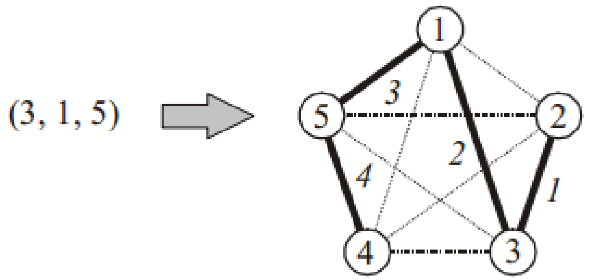


Algorytm otrzymywania drzewa z kodu Prufera

Dla zadanego ciągu liczb <a_1, a_2,..., a_{n-2}> wybranych w dowolny sposób ze zbioru {*1*,...,*n*}, aby wyznaczyć drzewo *T*, dla którego ciąg ten jest kodem Prufera, należy:

1. Zapisać dwie listy; pierwszą <*a1*, *a2*,..., a_{n-2}> oraz drugą *1*,*2*,...,*n* i rozpocząć ze zbiorem wierzchołków {*1*,...,*n*} i pustym zbiorem krawędzi.
2. Wyznaczyć z drugiej listy najmniejszą liczbę, powiedzmy *i*, która nie występuje na pierwszej liście. Liczbę *i* łączymy z liczbą z pierwszej listy np. *j* pierwszą z lewej. Usunąć pierwszy element z pierwszej listy, powiedzmy *j*, usunąć *i* z drugiej listy oraz dodać do zbioru krawędzi *ji*.
3. Jeśli pierwsza lista zawiera co najmniej jedną liczbę, to przejść do punktu 2. Jeśli pierwsza lista jest pusta, to druga będzie się składać z dokładnie dwóch liczb. Dodać do zbioru krawędzi ostatnią, której wierzchołkami są właśnie te liczby i zakończyć algorytm.

Przykład odkodowania drzewa: <~~3~~, ~~1~~, ~~5~~> oraz {~~1~~, ~~2~~, ~~3~~, 4, 5}

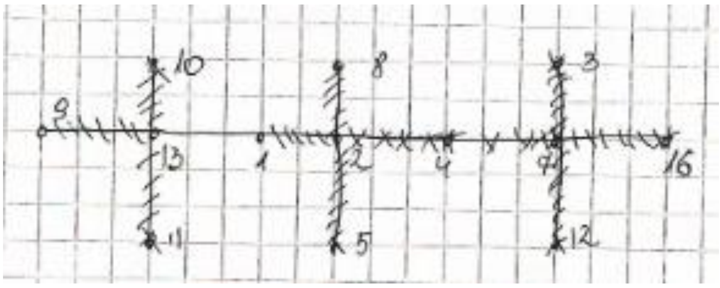


Kolejność kroków
1. Para (2,3), *ij* które usuwamy z list
2. Para (3,1), które usuwamy z listy
3. Para (1,5), które usuwamy z listy
4. Para (4,5) ostatnia w drugiej liscie
5. Rysowanie grafu na podstawie par krawędzi (*numery krawędzi oznaczają kolejne pary*)

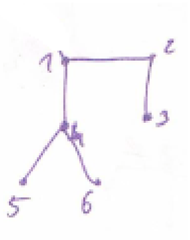
Kodowanie drzew oznakowanych

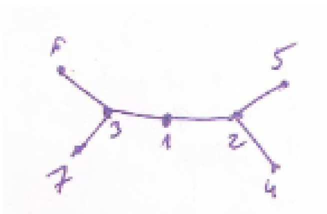
Każde oznakowane drzewo o wierzchołkach oznaczonych 1,2,…*n* można przedstawić za pomocą kodu [k_1, k_2, …, k_{n-2}] utworzonego następująco:

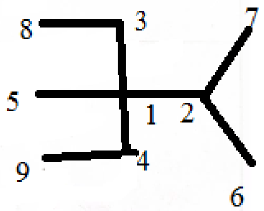
1. Szukamy wierzchołka stopnia 1 o najmniejszym numerze
2. Skreślamy go wraz z incydentną z nim krawędzią
3. Numer drugiego wierzchołka incydentnego ze skreśloną krawędzią notujemy jako numer kodu
4. Operacje powtarzamy dla pozostałego drzewa, procedurę kończymy w momencie otrzymania drzewa o jednej krawędzi



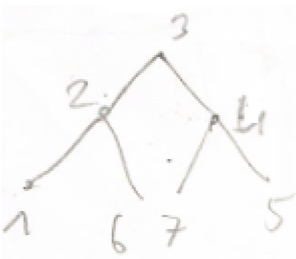
Kod Prufera drzewa dla drzewa T =<7, 2, 2, 13,13,13, 7, 1, 2, 4, 7>

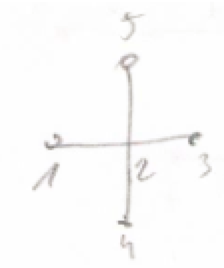


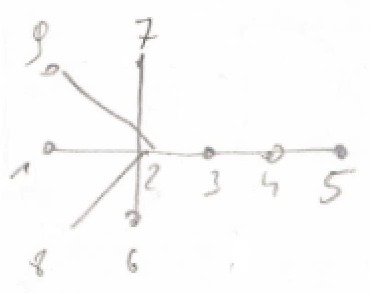




Kod Prufera <2, 1, 4, 4>   <2, 2, 1, 3, 3>   <1,2,2,1,3,1,4>







Kod Prufera <2, 4,2, 3, 4>   <2, 2, 2 >   <2, 4, 3, 2, 2, 2, 2>

Zadanie

Narysować drzewo na podstawie kodu prufera:

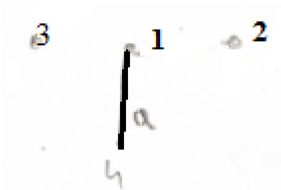
Kod <**1, 2, 3, 1, 2** > oraz

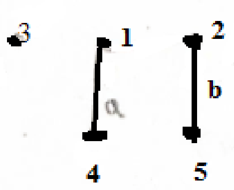
{1, 2, ~~3~~, 4, 5, 6, 7}lista wierzchołków

Kolejność kroków

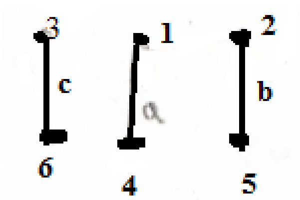
1. Para (4,1), *to krawędź a*
2. Para (5,2), *to krawędź b*
3. Para (6,3), *to krawędź c*
4. Para (3,1) *to krawędź d*
5. Para (1,2) *to krawędź e*
6. Ostatnia Para (2,7) *to krawędź f*

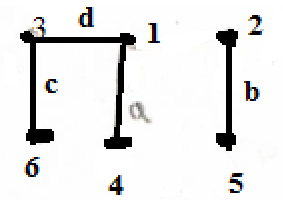
Krok 1   krok 2

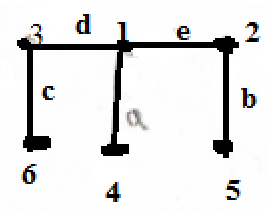


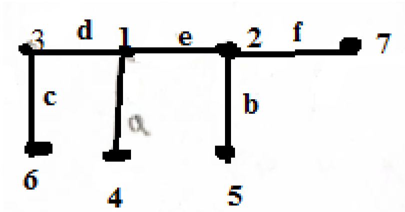


Krok 3   krok 4   Krok 5   krok 6 –> drzewo odkodowane

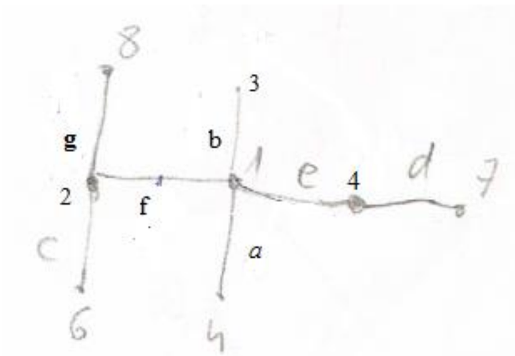


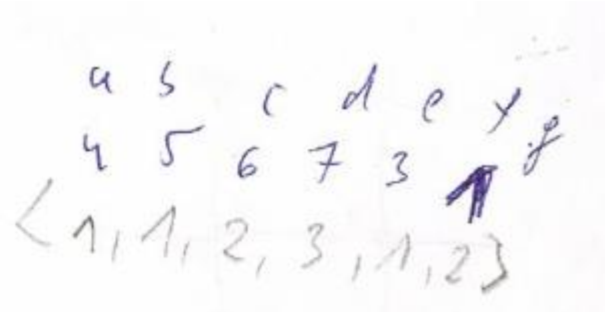






Narysować drzewo na podstawie kodu Prufera <1, 1, 2, 3, 1, 2>

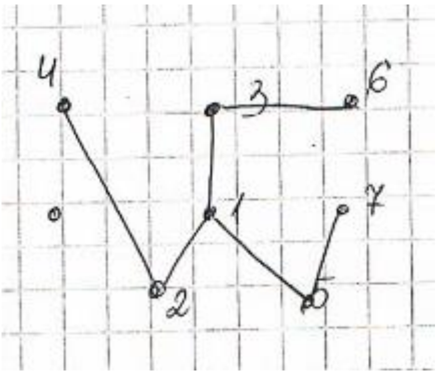


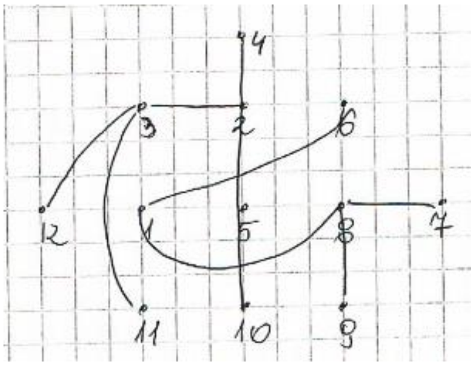


Narysować drzewo na podstawie kodu

<2,1,3,1,5> kod <2, 1, 8, 8, 1, 5, 5, 2, 3,3>

{1, 2, 3, 4, 5, 6, 7} lista 7 wierzchołków {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12} lista 12 wierzch.





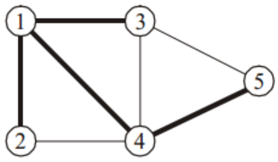
Drzewa rozpinające

Drzewo rozpinające grafu (*Spanning Tree*) zawiera wszystkie wierzchołki grafu *G*, zaś zbiór krawędzi drzewa jest podzbiorem zbioru krawędzi grafu.

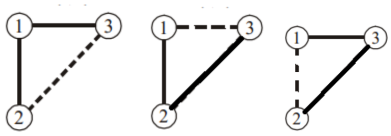
Dla grafu spójnego $G=(V, E)$ każde drzewa $G_T=(V, E)$ takie, że $T \subseteq E$ nazywamy ***drzewem rozpinającym*** grafu.

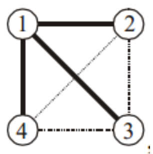
Konstrukcja **drzewa rozpinającego** polega na usuwaniu z grafu tych krawędzi, które należą do cykli. Najmniejszą liczbę krawędzi jaką trzeba usunąć z grafu, aby graf stał się acykliczny (stał się drzewem) nazywa się **rzędem acykliczności grafu** lub **liczbą cyklometryczną**.

Tw. Graf prosty ma drzewo rozpinające wtedy gdy jest spójny.

Przykład drzewa rozpinającego: 

Graf pełny K_n ma n^{n-2} drzew rozpinających.

Graf pełny K_3 *ma* $3^{3-2} = 3$ *drzewa rozpinające:* . 

Graf pełny K_4 ma $4^{4-2} = 4^2 = 16$ *drzew rozpinających. Np.* 

Pn, Sn – 1 drzewo

Cn ma *n* drzew rozpinających

Wyznaczanie liczby drzew rozpinających w dowolnych grafach spójnych

Twierdzenie Kirchhoffa (twierdzenie macierzowe o drzewach) to twierdzenie które pozwala wyznaczyć liczbę drzew rozpinających w grafie. Jest ono uogólnieniem wzoru Cayleya o liczbie drzew rozpinających w grafie pełnym.

Niech *G* będzie spójnym grafem nieskierowanym o *n* wierzchołkach Niech *M* będzie **laplasjanem** grafu, czyli macierzą $n \times n$, taką że:

$$a_{ij} = \begin{cases} \deg(v_i) & \text{dla } i = j \\ -1 & \text{dla } i \neq j \text{ oraz } v_i \text{ sasiaduje z } v_j \\ 0 & \text{w pozostalych przypadkach} \end{cases}$$

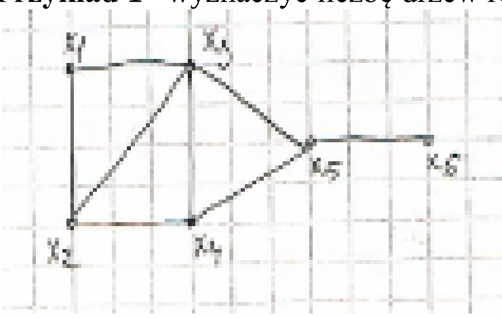
Wtedy liczba wszystkich drzew rozpinających grafu *G* będzie równa dopełnieniu algebraicznemu dowolnego wyrazu macierzy ***M***.

Dopełnienie algebraiczne elementu a_{ij} nazywamy liczbę D_{ij} określoną wzorem:

$D_{ij} = (-1)^{i+j} \det(A_{ij})$,

Gdzie A_{ij} = jest macierzą stopnia ***n*-1**, powstałą z macierzy *M* przez usunięcie *i*-tego wiersza oraz *j*-tej kolumny.

Przykład 1– wyznaczyć liczbę drzew rozpinających grafu **G1**:



Tworzymy Laplasjan *M* grafu G1:

$$M=\begin{bmatrix} 2 & -1 & -1 & 0 & 0 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 \\ -1 & -1 & 4 & -1 & -1 & 0 \\ 0 & -1 & -1 & 3 & -1 & 0 \\ 0 & 0 & -1 & -1 & 3 & -1 \\ 0 & 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

Wybieramy dowolny element macierzy *M* np. a_{61} i obliczmy jego dopełnienie algebraiczne D_{61}

Obliczamy dopełnienie elementu a_{11} czyli D_{11}

$$D_{61} = (-1)^{6+1} \begin{vmatrix} -1 & -1 & 0 & 0 & 0 \\ 3 & -1 & -1 & 0 & 0 \\ -1 & 4 & -1 & -1 & 0 \\ -1 & -1 & 3 & -1 & 0 \\ 0 & -1 & -1 & 3 & -1 \end{vmatrix} = (-1)(-1)(-1)^{5+5} \begin{vmatrix} -1 & -1 & 0 & 0 \\ 3 & -1 & -1 & 0 \\ -1 & 4 & -1 & -1 \\ -1 & -1 & 3 & -1 \end{vmatrix} =$$

$$W_4 = W_4 - W_3 = +1* \begin{vmatrix} -1 & -1 & 0 & 0 \\ 3 & -1 & -1 & 0 \\ -1 & 4 & -1 & -1 \\ 0 & -5 & 4 & 0 \end{vmatrix} = (-1)(-1)^{3+4} \begin{vmatrix} -1 & -1 & 0 \\ 3 & -1 & -1 \\ 0 & -5 & 4 \end{vmatrix} =$$

$$= 4 + 0 + 0 - (0 - 5 - 12) = \mathbf{21}.$$

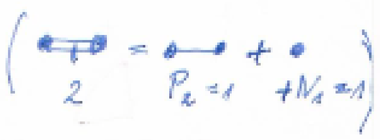
Graf G1 posiada 21 drzew rozpinających.

Metoda graficzna (wersja 1)

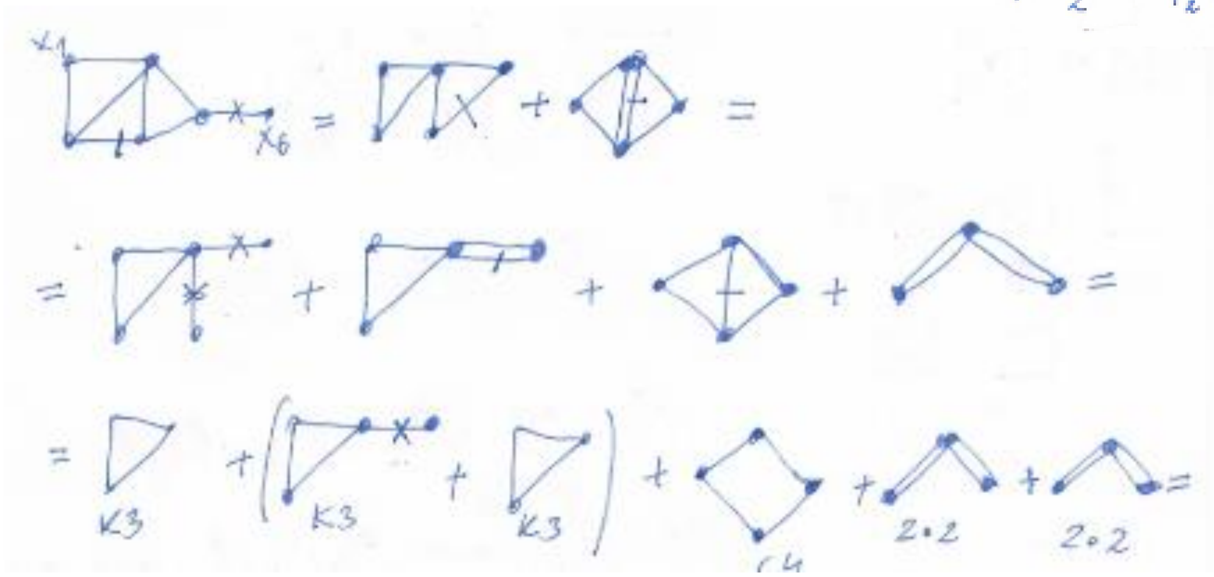
Metoda graficzna polega na usuwaniu krawędzi w grafie, w celu uzyskania grafu, dla którego zanana jest liczba drzew rozpinających.

Graf, z którego usuwany jest krawędź zastępowany jest równoważną **sumą dwóch innych grafów**. Pierwszy to graf z usuniętą krawędzią, a drugi to graf, w którym usuwana krawędź zwiera dwa incydentne wierzchołki (otrzymuje graf stopnia n-1). Pojedyncze krawędzie "doczepione" do cyklu w grafie są z definicji ignorowane, (pętle również są ignorowane)

NP. Redukcja podwójnej krawędzi:



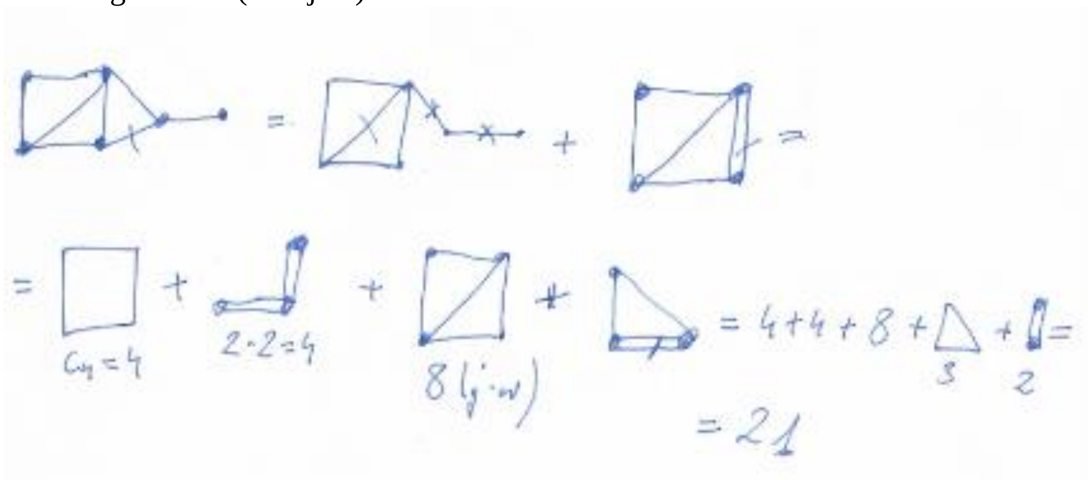
Graf G1



= 3 K_3 + C_4 + 2x2 + 2x2 = 9 +4+ 4 +4 = 21

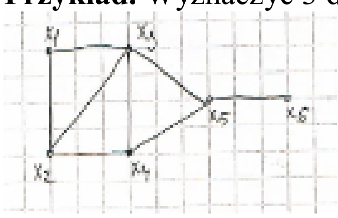
Liczba Drzew Rozpinających LDR= 3*3+ 4 + 4 + 4= 21

Metoda graficzna (wersja 2)

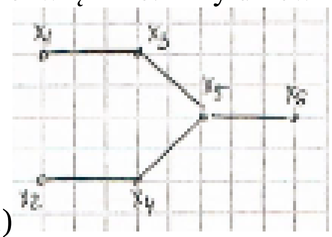


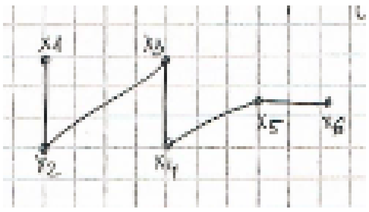
Liczba Drzew Rozpinających LDR= 21

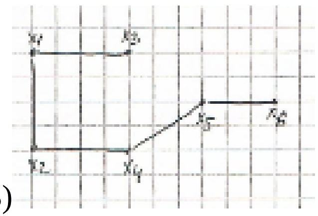
Przykład. Wyznaczyć 3 drzewa rozpinające grafu G1:



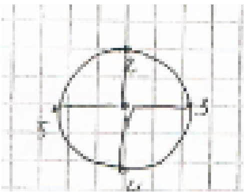
Rozwiązanie: Trzy drzewa (dwa drzewa ścieżkowe)

1) 

2) 

3) 

Przykład 2– wyznaczyć liczbę drzew rozpinających grafu **W5**:



Laplasjan grafu

$$M = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 4 & -1 & -1 & -1 & -1 \\ 2 & -1 & 3 & -1 & 0 & -1 \\ 3 & -1 & -1 & 3 & -1 & 0 \\ 4 & -1 & 0 & -1 & 3 & -1 \\ 5 & -1 & -1 & 0 & -1 & 3 \end{array}$$

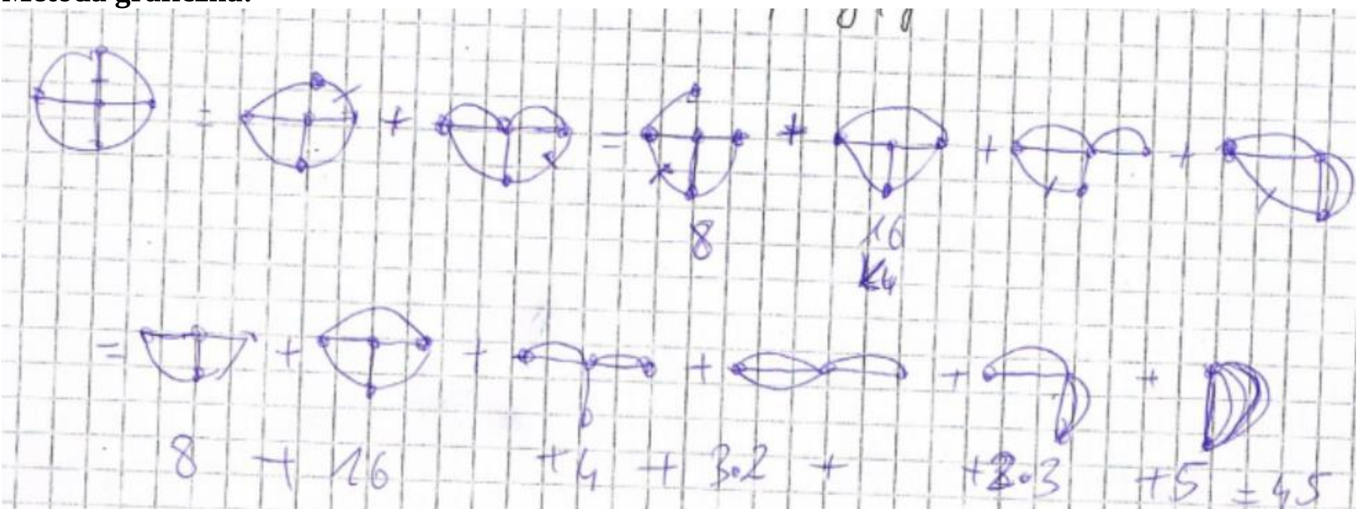
Obliczamy dopełnienie elementu a_{11} czyli D_{11}

$$D_{11} = (-1)^2 \begin{vmatrix} 3 & -1 & 0 & -1 \\ -1 & 3 & -1 & 0 \\ 0 & -1 & 3 & -1 \\ -1 & 0 & -1 & 3 \end{vmatrix} \underset{k_2 - k_4}{\overset{k_1+3k_4}{=}} \begin{vmatrix} 0 & 0 & 0 & -1 \\ -1 & 3 & -1 & 0 \\ -3 & 0 & 3 & -1 \\ 8 & -3 & -1 & 3 \end{vmatrix}$$

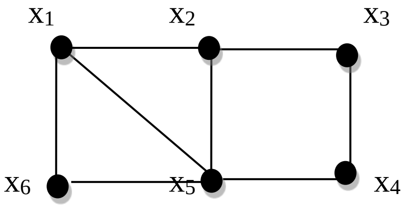
$$= (-1)\cdot(-1)^5 \begin{vmatrix} -1 & 3 & -1 \\ -3 & 0 & 3 \\ 8 & -3 & -1 \end{vmatrix} \begin{matrix} -1 & 3 & -1 \\ -3 & 0 & 3 \end{matrix} = 9+72-9-9 = 72-27 = 45$$

Graf W5 posiada 45 drzew rozpinających.

Metoda graficzna:



Przykład 3– wyznaczyć liczbę drzew rozpinających grafu **G3**:



Laplasjan M
$$M = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 2 & -1 & 0 & 0 & 0 & -1 \\ 2 & -1 & 4 & -1 & 0 & -1 & -1 \\ 3 & 0 & -1 & 2 & -1 & 0 & 0 \\ 4 & 0 & 0 & -1 & 2 & -1 & 0 \\ 5 & 0 & -1 & 0 & -1 & 3 & -1 \\ 6 & -1 & -1 & 0 & 0 & -1 & 3 \end{array}$$

Obliczamy dopełnienie elementu a_{11} czyli D_{11}

$$D_{11} = (-1)^2 \det \begin{bmatrix} 4 & -1 & 0 & -1 & -1 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & -1 & 2 & -1 & 0 \\ -1 & 0 & -1 & 3 & -1 \\ -1 & 0 & 0 & -1 & 3 \end{bmatrix} \underset{K_5 + 3K_1}{\overset{K_4 - K_1}{=}}$$

Wybieramy wiersze/kolumny gdzie są dwa zera

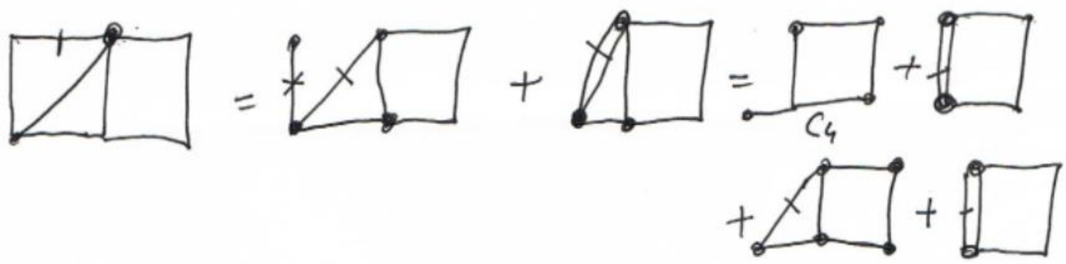
$$= \begin{vmatrix} 4 & -1 & 0 & -5 & 11 \\ -1 & 2 & -1 & 1 & -3 \\ 0 & -1 & 2 & -1 & 0 \\ -1 & 0 & -1 & 4 & -4 \\ -1 & 0 & 0 & 0 & 0 \end{vmatrix} = (-1)(-1)^{5+1} \begin{vmatrix} -1 & 0 & -5 & 11 \\ 2 & -1 & 1 & -3 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 4 & -4 \end{vmatrix} \Rightarrow K_3 = K_3 + K_4 =$$

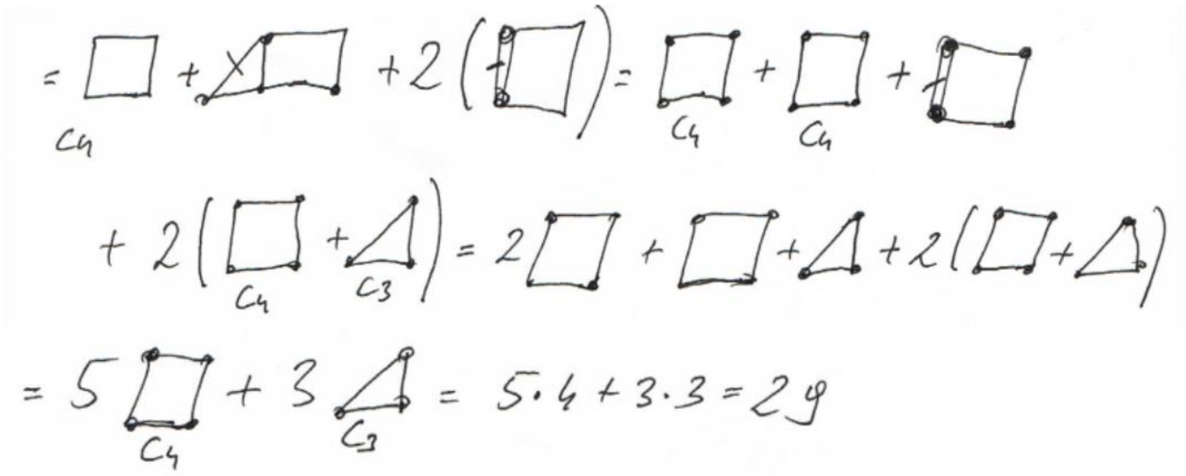
$$(-1)^* \begin{vmatrix} -1 & 0 & 6 & 11 \\ 2 & -1 & -2 & -3 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 0 & -4 \end{vmatrix} = K_4 = K_4 - 4K_2 = (-1) \begin{vmatrix} -1 & 0 & 6 & 11 \\ 2 & -1 & -2 & 1 \\ -1 & 2 & -1 & -8 \\ 0 & -1 & 0 & 0 \end{vmatrix} = (-1)(-1)(-1)^{4+2} \begin{vmatrix} -1 & 6 & 11 \\ 2 & -2 & 1 \\ -1 & -1 & -8 \end{vmatrix}$$

= -16 - 6 - 22 -22 -1 + 96 = - 67 + 96 = **29**

Graf G3 posiada **29** drzew rozpinających.

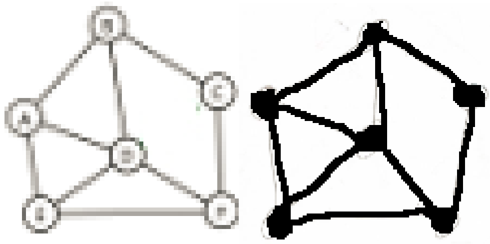
Metoda graficzna





Graf G3 posiada **29** drzew rozpinających.

Przykład 4– wyznaczyć liczbę drzew rozpinających grafu **G4**:



$$M=\begin{bmatrix} 3 & -1 & 0 & -1 & -1 & 0 \\ -1 & 3 & -1 & -1 & 0 & 0 \\ 0 & -1 & 2 & 0 & 0 & 0 \\ -1 & -1 & 0 & 4 & -1 & -1 \\ -1 & 0 & 0 & -1 & 3 & -1 \\ 0 & 0 & -1 & -1 & -1 & 3 \end{bmatrix}$$

Obliczamy dopełnienie elementu a_{11} czyli D_{11}

$$D_{11}=(-1)^{1+1}\begin{vmatrix} 3 & -1 & -1 & 0 & 0 \\ -1 & 2 & 0 & 0 & -1 \\ -1 & 0 & 4 & -1 & -1 \\ 0 & 0 & -1 & 3 & -1 \\ 0 & -1 & -1 & -1 & 3 \end{vmatrix}=\begin{matrix} K_1=K_1+3K_3 \\ K_2=K_2-K_3 \end{matrix}=$$

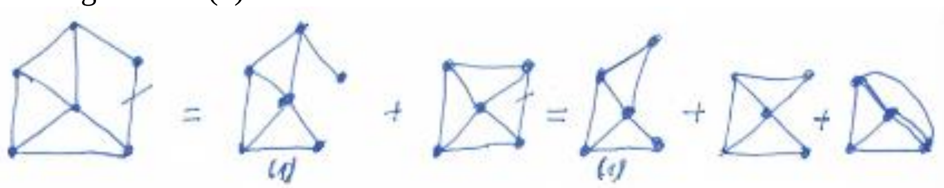
$$=1*\begin{vmatrix} 0 & 0 & -1 & 0 & 0 \\ -1 & 2 & 0 & 0 & -1 \\ 11 & -4 & 4 & -1 & -1 \\ -3 & +1 & -1 & 3 & -1 \\ -3 & 0 & -1 & -1 & 3 \end{vmatrix}=(-1)\,(-1)^{1+3}\begin{vmatrix} -1 & 2 & 0 & -1 \\ 11 & -4 & -1 & -1 \\ -3 & +1 & 3 & -1 \\ -3 & 0 & -1 & 3 \end{vmatrix}=\begin{matrix} K_1=K_1-K_4 \\ K_2=K_2+2K_4 \end{matrix}$$

$$=(-1)\begin{vmatrix} 0 & 0 & 0 & -1 \\ 12 & -6 & -1 & -1 \\ -2 & -1 & 3 & -1 \\ -6 & 6 & -1 & 3 \end{vmatrix}=(-1)(-1)(-1)^{1+4}\begin{vmatrix} 12 & -6 & -1 \\ -2 & -1 & 3 \\ -6 & 6 & -1 \end{vmatrix}=$$

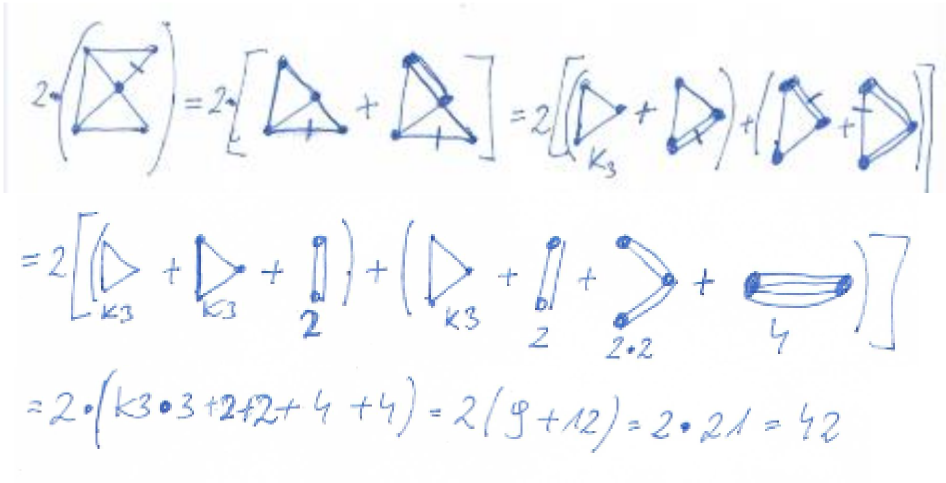
$(-1)^7(12+108+12+6+12-216)=(-1)(150-216)=66$

Graf G4 posiada 66 drzew rozpinających.

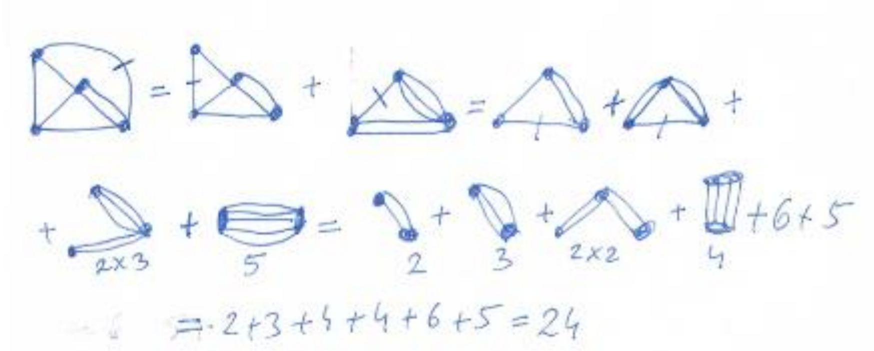
Metoda graficzna (1)



Gdzie:

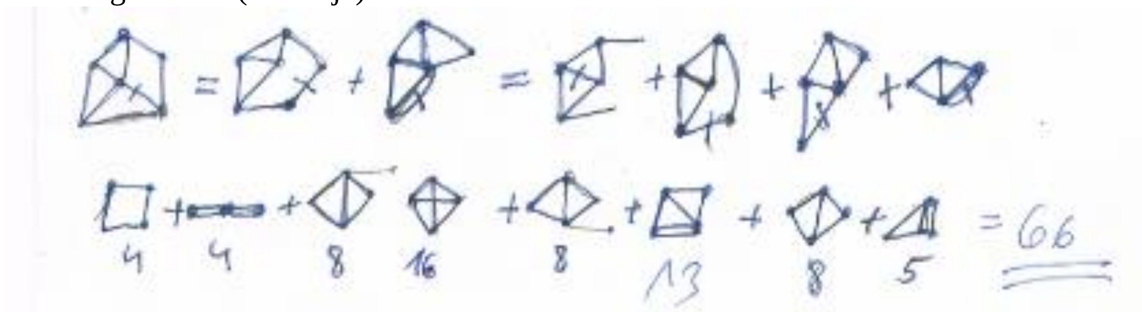


oraz



Liczba Drzew Rozpinających LDR= 42+ 24 = **66**

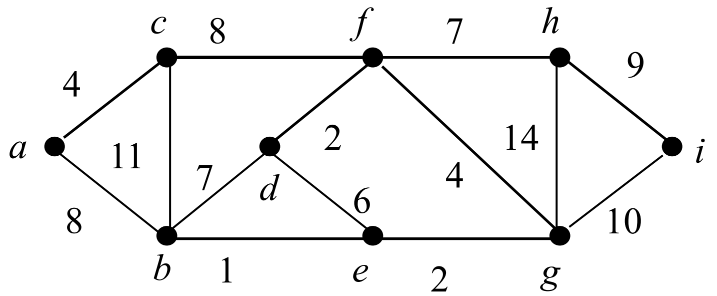
Metoda graficzna (2 wersja)



Liczba Drzew Rozpinających LDR= 42+ 24 = **66**

Grafy ważone - **Minimalne drzewa rozpinające**

Podgraf T spójnego grafu G nazywa się jego *drzewem rozpinającym*, jeśli T jest acykliczny i łączy wszystkie wierzchołki G. Jeśli krawędziom przypisane są wagi, i suma wag krawędzi drzewa T jest minimalna, T nazywamy ***minimalnym drzewem rozpinającym***. *(mst)*



Minimalne drzewo rozpinające (ang. *MST*, *minimum spanning tree*) – drzewo rozpinające danego grafu o najmniejszej z możliwych sumie wag, tj. takie, że nie istnieje dla tego grafu inne drzewo rozpinające o mniejszej sumie wag krawędzi. Kosztem grafu nazywamy sumę kosztów jego krawędzi
Dany jest graf ważony *G(V, E, w)*, gdzie **V** – zbiór wierzchołków, *E* – zbiór krawędzi, *w* – funkcja przypisująca krawędzi E_i wagę (liczbę rzeczywistą lub całkowitą).
Minimalnym drzewem rozpinającym nazywamy drzewo rozpinające T, dla którego suma wag

$$\sum_{e \in T} w(e)$$

jest najmniejsza z możliwych.
Dla niektórych grafów można wskazać wiele drzew rozpinających spełniających tę własność.

Istnieje kilka algorytmów do wyznaczania minimalnego drzewa rozpinającego grafu.

Jeżeli mamy do czynienia z grafem z wagami, to najczęściej interesuje nas znalezienie drzewa rozpinającego o minimalnej wadze, tzn., drzewa z najmniejszą sumą wag jego krawędzi.
Aby znaleźć drzewo o żądanych własnościach można zastosować dwa algorytmy:

- Kruskala (algorytm zachłanny)
- Prima (algorytm najbliższego sąsiada)

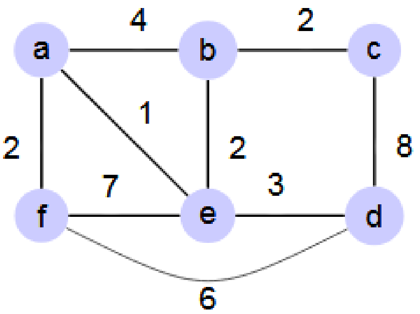
Algorytm Kruskala Minimalne drzewa rozpinające

Algorytm Kruskala jest oparty o metodę zachłanną i polega na złączeniu wielu poddrzew w jedno za pomocą krawędzi o najmniejszej wadze.

Problem. W spójnym grafie ważonym znaleźć najtańsze drzewo rozpinające.
Rozwiązanie. **Algorytm Kruskala.**
Wstępnym krokiem jest sortowanie niemalejące wszystkich krawędzi
S=∅;
i=1;
while (istnieje krawędź *e* taka, że graf indukowany przez S∪{*e*} nie zawiera cyklu)
{e_i = najtańsza krawędź nie należąca do S taka, że graf indukowany przez S∪{e_i} nie zawiera cyklu;
S=S∪{e_i};
i=i+1; }
return S;

- Wybierz krawędź (która nie jest pętlą) e_1 tak, by waga tej krawędzi była najmniejsza.
- Jeżeli krawędzie $e_1, e_2, \ldots, e_k$ zostały już wybrane, to z pozostałych $E \setminus \{e_1, e_2, \ldots, e_k\}$ wybierz krawędź e_{k+1} w taki sposób aby:
  - graf, który składa się tylko z krawędzi $e_1, e_2, \ldots, e_k, e_{k+1}$ był acykliczny, oraz
  - waga krawędzi e_{k+1} była najmniejsza.
- Jeśli nie można wykonać kroku 2, to STOP.

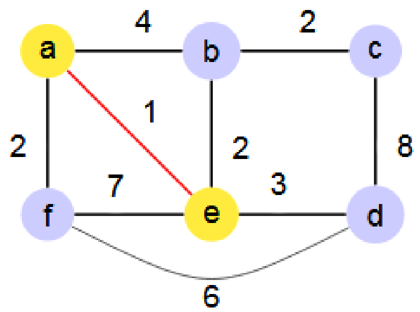
Przykład. Wyznaczyć minimalne drzewo rozpinające algorytmem Kruskala grafu G9



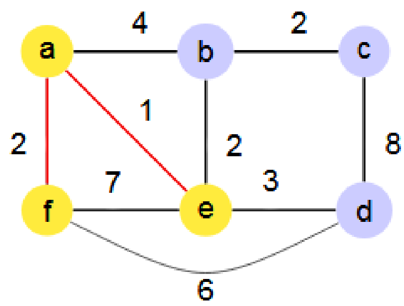
Na wstępie sortowanie krawędzi wg.wag
i otrzymujemy : ~~ae=1~~ , ~~af=2 , bc=2 , be=2~~, de=3 , ab=4 , fd=6 , ef=7, cd=8
Dodajemy do drzewa krawędzie z listy posortowanie, tak aby nowe krawędzie nie tworzyły cykli. Zaczynamy do krawędzi o najmniejszej wadze
Algorytm Kruskala

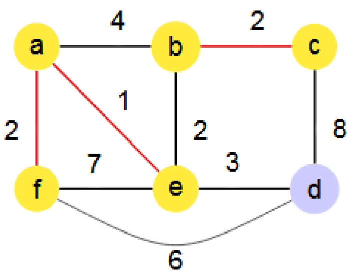
Lp	Krawędzie drzewa S	Zbiory wierzchołków
Krok 0	----	{a, b, c, d, e, f}
Krok 1	ae=1	{ ae, b, c, d , f}
Krok 2	ae, af=2	{ aef, b, c, d}
Krok 3	ae, af, bc=2	{ aef, bc, d}
Krok 4	ae, af, bc, be=2	{ aefbc, d}
Krok 5	ae, af, bc, be, de=3	{ aefbcd }

– Krok 1

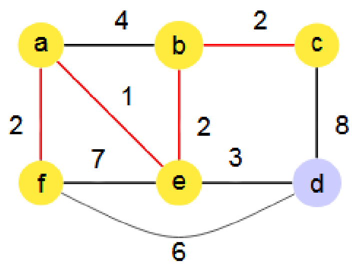


Krok 2

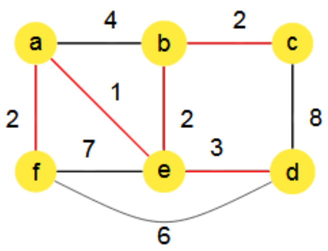




Krok 3

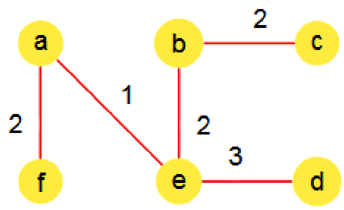


Krok 4



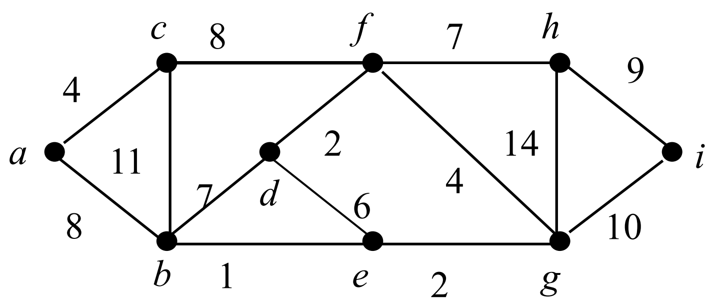
Krok 5

Minimalne Drzewo



Koszt minimalnego drzewa rozpinającego KMDR**= 2+1+2+2+3 = 10**

Przykład. Wyznaczyć minimalne drzewo rozpinające algorytmem Kruskala grafu G10

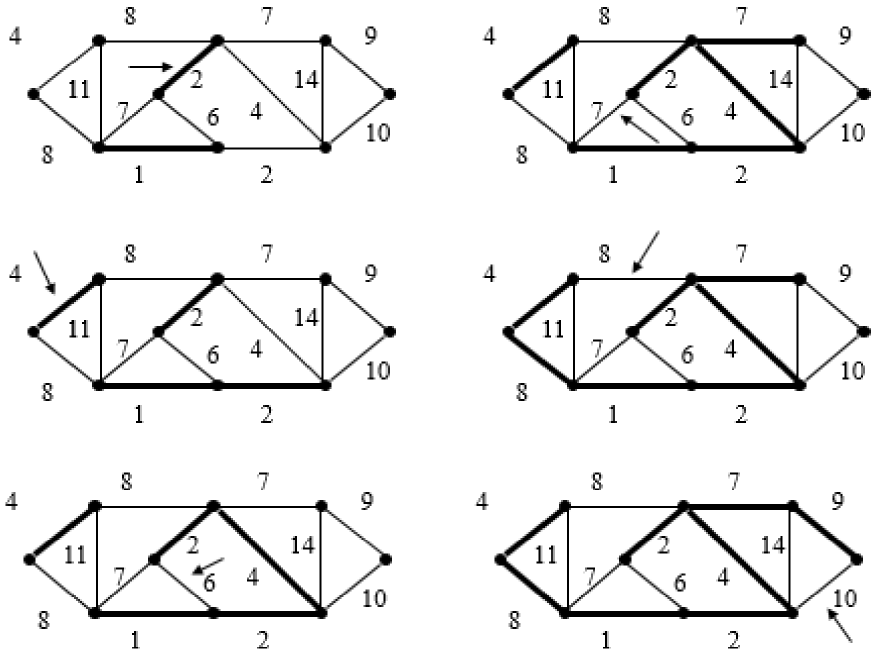


Na wstępie sortowanie krawędzi wg.wag
i otrzymujemy :be=1 , eg=2 , df=2 , ac=4 ,fg=4 , ~~de=6~~ , fh=7 ~~, bd~~=7, ab=8, ~~cf=8~~, hi=9, gi=10, gh=14

Dodajemy do drzewa krawędzie z listy posortowanie, tak aby nowe krawędzie nie tworzyły cykli. Zaczynamy do krawędzi o najmniejszej wadze
Algorytm Kruskala

Lp	Krawędzie drzewa S	Zbiory wierzchołków
Krok 0		{a, b, c, d, e, f, g, h, i}
Krok 1	be=1	{ a, be, c, d , f, g, h, i}
Krok 2	df=2	{ a, be, c, df, g, h, i}
Krok 3	eg=2	{ a, beg, c, df, h, i}
Krok 4	ac=4	{ ac, beg, df, h, i}
Krok 5	fg=4	{ ac, begdf, h, i}
Krok 6	fh=7	{ ac, begdfh, i}
Krok 7	ab=8	{ acbegdfh, i}
Krok 8	hi=9	{ acbegdfhi }

Postać drzewa S w kolejnych krokach algorytmu (niektóre rys. przedstawiają dwa kroki)

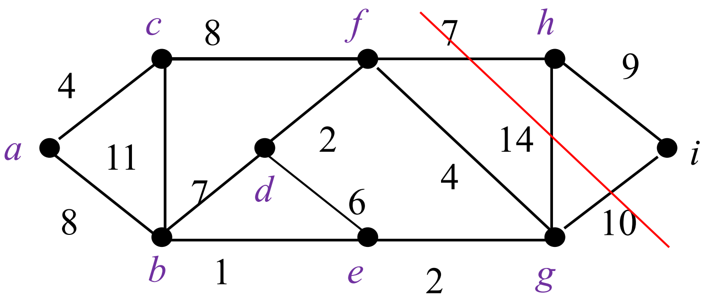


Algorytm Prima

Algorytm Prima jest, podobnie jak algorytm Kruskala oparty o metodę zachłanną.

- Budowę minimalnego drzewa rozpinającego zaczynamy od dowolnego wierzchołka, np. od pierwszego.
- Dodajemy wierzchołek do drzewa, a wszystkie krawędzie incydentne umieszczamy na posortowanej wg. wag liście.(krawędzie przekroju V\S)
- Następnie zdejmujemy z listy pierwszy element (o najmniejszej wadze==krawędź lekka) i jeżeli wierzchołek, który łączy nie należy do drzewa, dodajemy go do drzewa, a na liście znów umieszczamy wszystkie krawędzie incydentne z wierzchołkiem, który dodaliśmy.
- Jednym zdaniem: zawsze dodajemy do drzewa krawędź o najmniejszej wadze, osiągalną z jakiegoś wierzchołka tego poddrzewa.

Wyznacz minimalne drzewo rozpinające algorytmem Prima grafu G10



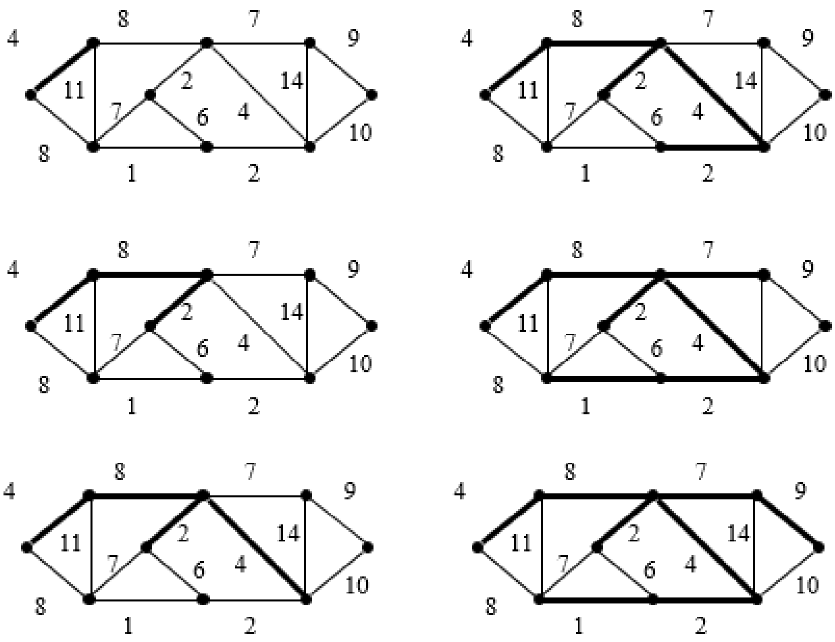
Algorytm Prima

Lp	Zbiór S	Zbiór V\S	Krawędzie przekroju { S, V\S }	Krawędź lekka
0		*a, b,c,d,e,f,g,h,i*	-	-
1	*a*	*b, c, d, e, f, g, h, i*	*ac*=4, *ab*=8	*ac*=4
2	*a, c*	*b, d, e, f, g, h, i*	*ab*=8, *cf*=8, *cb*=11	*cf*=8
3	*a, c, f*	*b, d, e, g, h, i*	*fd*=2, *fg*=4, *fh*=7, *ab*=8, *cb*=11	*fd*=2
4	*a, c, f, d*	*b, e, g, h, i*	*fg*=4, *de*=6, *db*=7, *fh*=7, *ab*=8, *cb*=11	*fg*=4
5	*a, c, f, d, g*	*b, e, h, i*	*eg*=2, *de*=6, *db*=7, *fh*=7, *ab*=8, *gi*=10, *cb*=11, *gh*=14,	*eg*=2

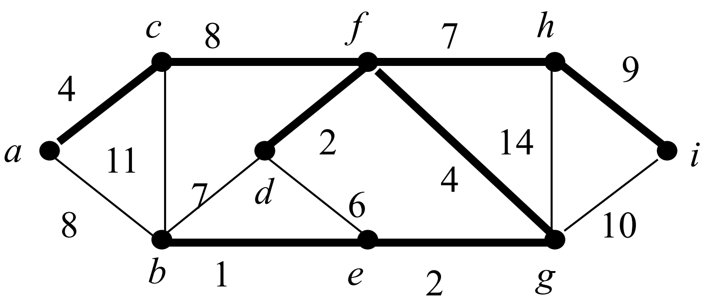
6	*a, c, f, d, g, e*	*b, h, i*	*eb*=1, *db*=7, *fh*=7, *ab*=8, *gi*=10, *cb*=11 *gh*=14,	*eb*=1
7	*a, c, f, d, g, e, b*	*h, i*	*fh*=7, *gi*=10, *gh*=14	*fh*=7
8	*a,c,f,d,g, e, b,h*	*i*	*hi*=9, *gi*=10	*hi*=9
9	*a,c,f,d,g,e,b,h,i*	--	--	*Koszt* =**37**

Koszt minimalnego drzewa rozpinającego KMDR**= 4+8+2+4+2+1+7+ 9= 37**

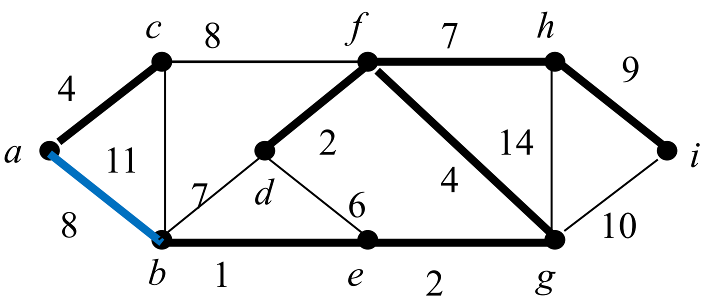
Kolejne kroki algorytmu Prima



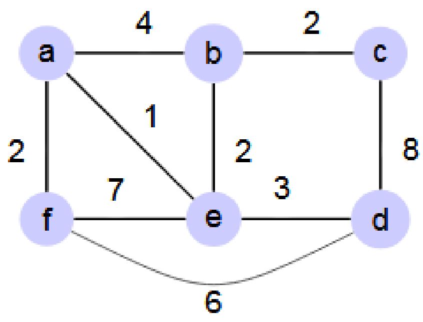
Minimalne drzewo rozpinający grafu G10



Graf G10 posiada 2 minimalne drzewa o koszcie 37, ponieważ krawędź cf=8 można zastąpić krawędzią ab=8. ()



Przykład. Wyznaczyć minimalne drzewo rozpinające algorytmem Prima G9



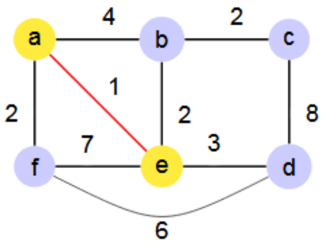
Algorytm Prima

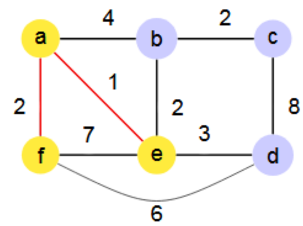
Lp	Zbiór S	Zbiór V\S	Krawędzie przekroju {S, V\S}	Krawędź lekka
0		*a, b, c, d, e, f*	-	-
1	*a*	*b, c, d, e, f*	*ae*=1, *af*=2, *ab*=4,	*ae*=1
2	*a, e*	*b, c, d, f*	*af*=2, *eb*=2, *ed*=3, *ab*=4, *ef*=7	*af*=2
3	*a, e, f*	*b, c, d*	*eb*=2, *ed*=3, *ab*=4, *fd*=6	*eb*=2
4	*a, e, f, b*	*c, d*	*bc*=2, *ed*=3, *fd*=6	*bc*=2
5	*a, e, f, b, c*	*d*	*ed*=3, *fd*=6, *cd*=8	*ed*=3
6	*a, e, f, b, c, d*	--	--	Koszt =**10**

Koszt minimalnego drzewa rozpinającego KMDR**= 1+2+2+2+3 = 10**

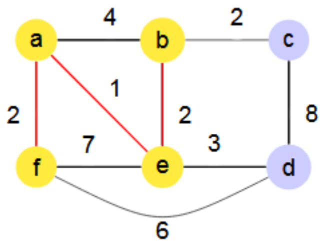
Kolejne kroki algorytmu Prima

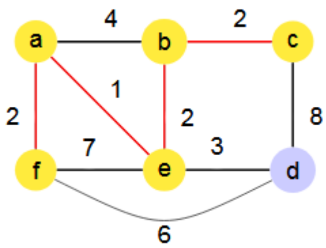
Krok 1





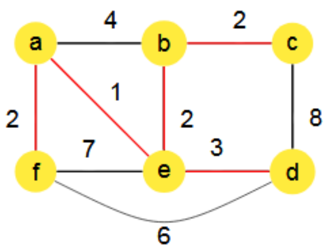
Krok 2





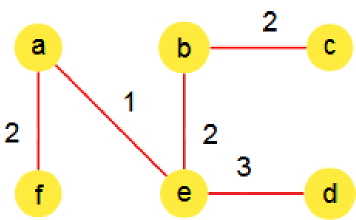
Krok 3

Krok 4



Krok 5

Minimalne Drzewo



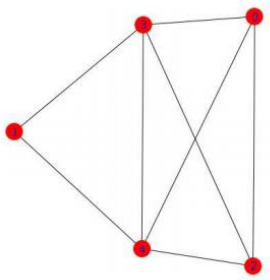
Zadania

- Można wykonać po jednym podpunkcie z każdego zadania. (*do odesłania*)
- Można wykonać dowolną liczbę zadań. (*nie odsyłać*)

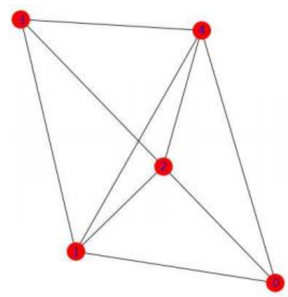
Zadanie1. Narysować drzewo na podstawie kodu Prufera

A) <2,1,3,2,4> B) <1, 2, 3, 4, 5, 5, 4, 3,2, 1> C) <1, 1, 2, 2, 5, 5, 3>
D) <5,1,4,2,1,3,6> E) <1, 2, 1, 3 ,1,4,1,5> F) <1,2,3,2,3,1,3>

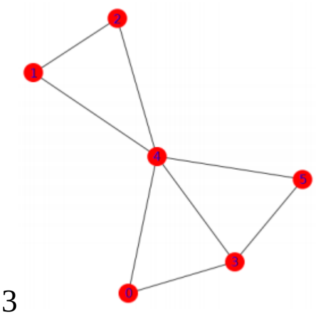
Zadanie 2. Wyznacz liczbę drzew rozpinających następujących grafów (metoda macierzowa / metoda graficzna)



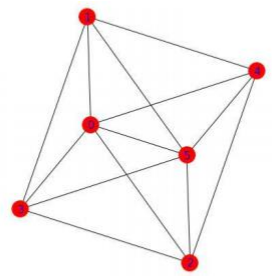
Graf 21



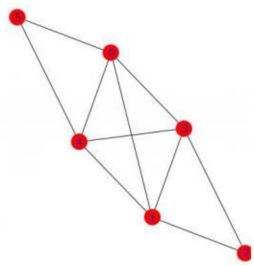
Graf 22



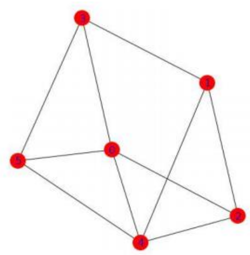
Graf 23



Graf 24

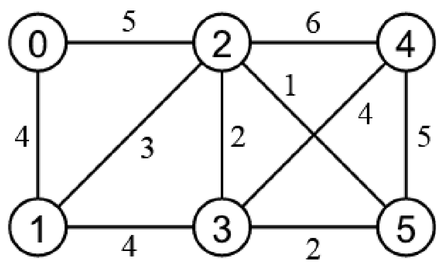


Graf 25

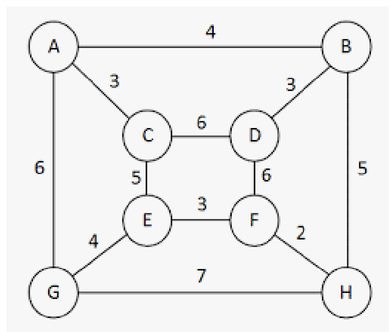


Graf 26

Zadanie. Wyznaczyć minimalne drzewo rozpinające algorytmem Kruskala i Prima

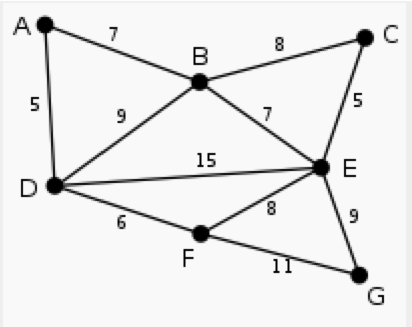


Graf G31



Graf G32

Graf G33



Graf G34