

4c. Drzewa spinające

Grzegorz Kosiorowski

Uniwersytet Ekonomiczny w Krakowie

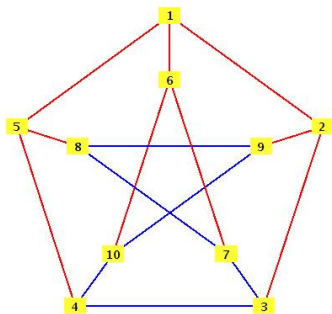
Drzewo spinające

Drzewem spinającym (rozpinającym) grafu G nazywamy podgraf G zawierający wszystkie jego wierzchołki i będący drzewem.

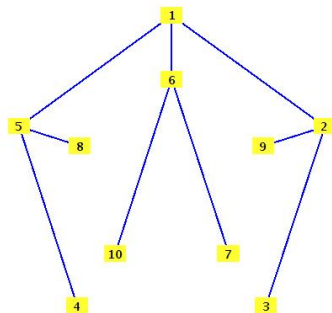
Drzewo spinające jest najmniejszym (w sensie liczby krawędzi, jak również zawierania zbioru krawędzi) podgrafem spójnym łączącym wszystkie wierzchołki grafu G . Dlatego wyznaczanie drzew spinających jest przydatne, gdy wierzchołki grafu symbolizują cele, jakie chcemy osiągnąć: drzewa spinające wskażą najmniejszy „koszt”, którym możemy to zrobić.

Drzewo spinające - przykład

Przykładowe drzewo spinające w grafie Petersena:



Tu na czerwono.



A tu bez reszty grafu.

O istnieniu drzewa spinającego

Każdy skończony graf ma drzewo spinające wtedy i tylko wtedy, gdy jest spójny.

Oczywiście, zazwyczaj w grafie jest wiele możliwych drzew spinających.

Algorytm znajdowania drzewa spinającego, jest tak prosty, jak tylko można sobie wymarzyć, o ile nie mamy innych wymagań co do postaci tego drzewa.

Algorytm wyznaczania drzewa spinającego

Algorytm wyznaczania drzewa spinającego

Dane: Graf spójny skończony $G = (V(G), E(G))$, v - wierzchołek grafu spójnego G .

Zmienne: E - zbiór krawędzi, V - zbiór wierzchołków.

- I. Niech $V := \{v\}$, $E := \emptyset$.
- II. Dopóki istnieją w grafie G krawędzie łączące wierzchołki ze zbioru V z wierzchołkami, które nie należą do V : wybierz krawędź uw łączącą wierzchołek $u \in V$ z wierzchołkiem $w \notin V$, dołącz w do V i dołącz uw do E .
- **Rezultat:** E to zbiór krawędzi drzewa spinającego grafu G .

Uwagi na temat algorytmu wyznaczania drzewa spinającego

- Algorytm działa wyjątkowo prosto - startujemy z dowolnego wierzchołka i dokładamy kolejne krawędzie, łącząc już stworzoną część nowego grafu z kolejnymi wierzchołkami, które nie zostały jeszcze „podłączone” do naszego drzewa, aż nie pozostanie ani jeden niepołączony wierzchołek lub skończą się krawędzie łączące wierzchołki połączone z niepołączonymi.
- Tego algorytmu można też użyć do badania, czy graf jest spójny (np. jako kroku badającego czy krawędź jest mostem w algorytmie Fleury’ego). Jeśli algorytm skończy działanie i zbiór V nie zawiera wszystkich wierzchołków G , to G jest niespójny.
- Czas działania tego algorytmu to $O(|V(G)| + |E(G)|)$.

Przykłady zastosowań drzew spinających

- Wyznaczanie minimalnej koniecznej struktury (np. minimalnej sieci wodociągowej docierającej do wszystkich domów w miejscowości).
- Broadcasting - wyznaczenie dozwolonych ścieżek po których sygnał ma iść od nadajnika do odbiorców, tak, by nie docierał do nich kilkakrotnie (unikanie redundancji).

Minimalne drzewo spinające

Drzewa spinające są przydatne, ale by zoptymalizować ich użyteczność, czasem potrzeba czegoś więcej. Na przykład, rozważając konieczną do zaopatrzenia wszystkich domów w miejscowości sieć wodociągową, możemy zażądać, by była ona jak najkrótsza (w sensie całkowitej długości rur). Wtedy poszukujemy tzw. *minimalnego drzewa spinającego*.

Minimalne drzewo spinające

W grafie G z wagami *minimalnym drzewem spinającym* nazywamy takie drzewo, które jest spinające i jego waga (czyli suma wag jego krawędzi) jest nie większa niż waga jakiegokolwiek innego drzewa spinającego w tym samym grafie.

Na szczęście, również wyznaczanie minimalnych drzew spinających jest dość proste. Działają tutaj dwa algorytmy zachłanne: algorytm Kruskala i algorytm Prima.

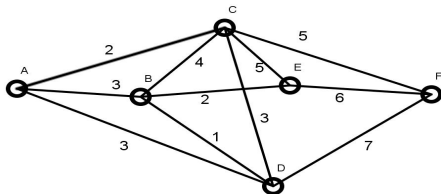
Algorytm Kruskala

Dane: G - skończony graf spójny z wagami, którego krawędzie są uporządkowane wg wzrastających wag e_1, \dots, e_n , $n = |E(G)|$.

Zmienne: E - zbiór krawędzi.

- I. Podstaw $E := \emptyset$.
- II. Dla $j = 1$ do n wykonuj: jeśli graf $E \cup \{e_j\}$ jest acykliczny, dołącz e_j do E .
- **Rezultat:** E to zbiór krawędzi minimalnego drzewa spinającego G .

Algorytm Kruskala - przykład

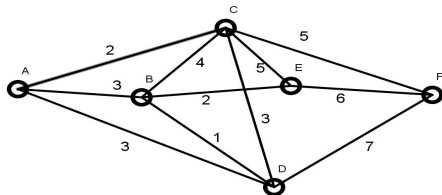


Należy znaleźć minimalne drzewo spinające w grafie powyżej za pomocą algorytmu Kruskala.

Przygotowanie do algorytmu wymaga wypisania krawędzi w kolejności wzrastających wag (jeśli wagi dwu krawędzi są równe, to ich kolejność jest dowolna).

W tym przypadku to uporządkowanie może wyglądać np. tak: BD, AC, BE, AB, AD, CD, BC, CE, CF, EF, DF.

Algorytm Kruskala - przykład

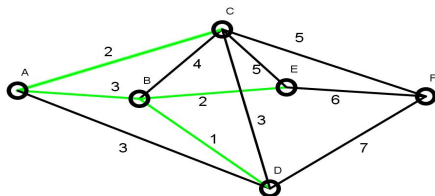


Kolejność krawędzi: BD, AC, BE, AB, AD, CD, BC, CE, CF, EF, DF.

Działanie algorytmu zapisujemy w takiej tabelce:

Nr kroku	wybrana krawędź	krawędzie odrzucone przed wyborem
1	?	?
2	?	?

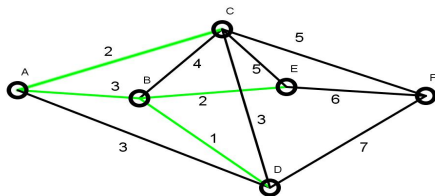
Algorytm Kruskala - przykład



Dołączenie pierwszych 4 krawędzi: BD,AC,BE,AB nie sprawia żadnych problemów.

Nr kroku	wybrana krawędź	krawędzie odrzucone przed wyborem
1	BD	-
2	AC	-
3	BE	-
4	AB	-

Algorytm Kruskala - przykład



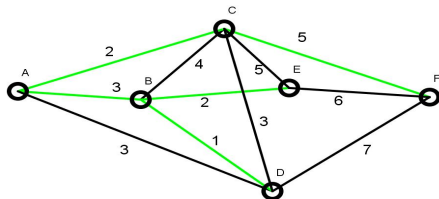
Kolejność krawędzi: $BD, AC, BE, AB, AD, CD, BC, CE, CF, EF, DF$.

Nie możemy teraz dołączyć krawędzi AD , gdyż stworzyłoby to cykl $ABDA$.

Krawędź CD również odpada (cykl $ACDBA$), podobnie BC (cykl $ABCA$), CE tworzy cykl $ACEBA$. Dopiero krawędź CF pasuje.

Jednocześnie, dołączenie tej krawędzi powoduje, że wszystkie wierzchołki grafu są połączone, więc otrzymaliśmy nasze drzewo spinające (dołączenie dowolnej z następnych krawędzi musiałoby stworzyć cykl).

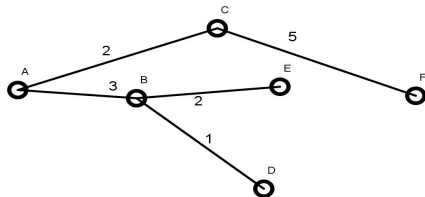
Algorytm Kruskala - przykład



Działanie algorytmu opisuje tabelka:

Nr kroku	wybrana krawędź	krawędzie odrzucone przed wyborem
1	BD	-
2	AC	-
3	BE	-
4	AB	-
5	CF	AD,CD,BC,CE

Algorytm Kruskala - przykład



Wynikiem algorytmu jest powyższe minimalne drzewo spinające.
Waga tego drzewa wynosi 13 (suma wag krawędzi).

Algorytm Kruskala - uwagi

- Algorytm Kruskala działa poprawnie, nawet jeśli graf G ma pętle i krawędzie wielokrotne.
- Nie trzeba nawet zakładać spójności - jednak wtedy znajdziemy nie drzewo, a las spinający.
- Wadą tego algorytmu jest to, że przed jego zastosowaniem musimy jakoś uporządkować krawędzie według wzrastających wag.
- Jednakże, nawet razem z tym uporządkowaniem, algorytm Kruskala zajmuje tylko $O(|E(G)| \cdot \log |E(G)|)$ czasu.
- Problemem może być jedynie sytuacja, gdy na początku tego uporządkowania krawędzi nie da się uzyskać (np. dane o wagach uzyskujemy w miarę tworzenia się naszego drzewa). W takiej sytuacji lepiej zastosować algorytm Prima.

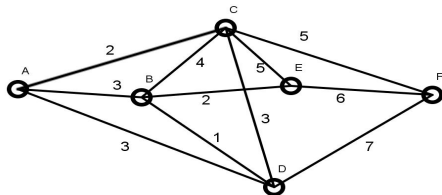
Algorytm Prima

Dane: G - skończony graf spójny z wagami.

Zmienne: E - zbiór krawędzi, V - zbiór wierzchołków.

- I. Podstaw $E := \emptyset$.
- II. Wybierz w - dowolny wierzchołek ze zbioru $V(G)$ i niech $V := \{w\}$.
- III. Dopóki $V \neq V(G)$ wykonuj: wybierz w zbiorze $E(G)$ krawędź uv o najmniejszej możliwej wadze, taką, że $u \in V$ i $v \in V(G) \setminus V$, a następnie dołącz krawędź uv do zbioru E i wierzchołek v do zbioru V .
- **Rezultat:** E to zbiór krawędzi minimalnego drzewa spinającego G .

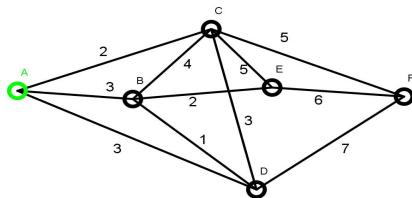
Algorytm Prima - przykład



Będziemy wybierać krawędzie do drzewa, zaczynając od wierzchołka A. Kolejne kroki algorytmu zapisujemy w takiej tabeli:

Nr etapu	wybór krawędzi	alternatywy, dozwolone wybory
1	?	?
2	?	?

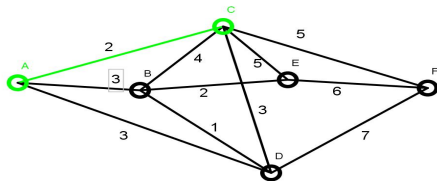
Algorytm Prima - przykład



$V = \{A\}$. Możliwe połączenia między V , a $V(G) \setminus V$ to AB , AC i AD . Z nich, najmniejszą wagę ma AC , więc dołączamy AC do drzewa, a C do zbioru V .

Nr etapu	wyбір krawędzi	alternatywne, dozwolone wybory
1	AC	-

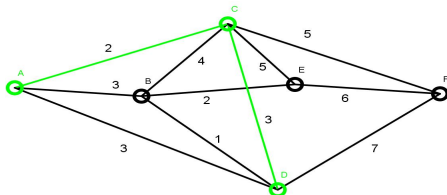
Algorytm Prima - przykład



$V = \{A, C\}$. Możliwe połączenia między V , a $V(G) \setminus V$ to AB , AD , CB , CD , CE , CF . Z nich, najmniejsze wagi mają AB , AD i CD . Wybieramy dowolną z nich (np. CD) dołączając ją do drzewa, drugi jej koniec (czyli D) dołączamy do V , a pozostałe z tych krawędzi zaznaczamy w tabelce jako alternatywne, dozwolone wybory.

Nr etapu	wybór krawędzi	alternatywne, dozwolone wybory
2	CD	AB,AD

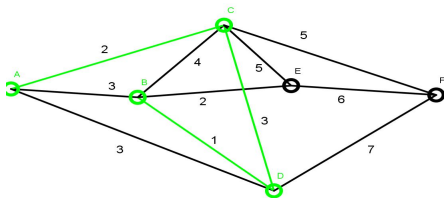
Algorytm Prima - przykład



$V = \{A, C, D\}$. Możliwe połączenia między V , a $V(G) \setminus V$ to wszystkie czarne krawędzie powyższego grafu poza AD , BE i EF . Z nich, najmniejszą wagę ma BD , więc dołączamy BD do drzewa, a B do zbioru V .

Nr etapu	wybór krawędzi	alternatywne, dozwolone wybory
3	BD	-

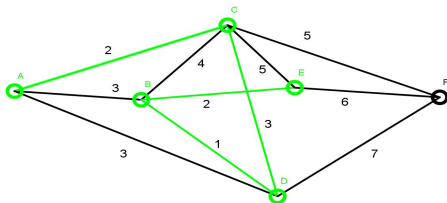
Algorytm Prima - przykład



$V = \{A, B, C, D\}$. Możliwe połączenia między V , a $V(G) \setminus V$ to BE , CE , CF i DF . Z nich, najmniejszą wagę ma BE , więc dołączamy BE do drzewa, a E do zbioru V .

Nr etapu	wyбір krawędzi	alternatywne, dozwolone wybory
4	BE	-

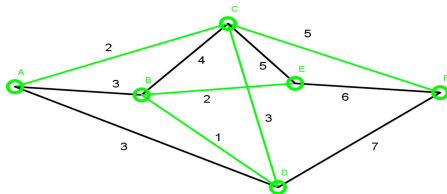
Algorytm Prima - przykład



$V = \{A, B, C, D, E\}$. Możliwe połączenia między V , a $V(G) \setminus V$ to CF i DF . Z nich, najmniejszą wagę ma CF , więc dołączamy CF do drzewa, a F do zbioru V .

Nr etapu	wyбір krawędzi	alternatywne, dozwolone wybory
5	CF	-

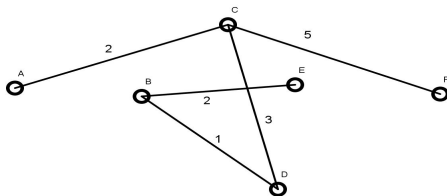
Algorytm Prima - przykład



$V = \{A, B, C, D, E, F\} = V(G)$, więc algorytm się kończy. Działanie algorytmu Prima opisuje tabela:

Nr etapu	wybór krawędzi	alternatywne, dozwolone wybory
1	AC	-
2	CD	AB, AD
3	BD	-
4	BE	-
5	CF	-

Algorytm Prima - przykład



Wynikiem algorytmu jest powyższe minimalne drzewo spinające. Waga tego drzewa wynosi 13 (suma wag krawędzi).

Warto zauważyć, że minimalne drzewo spinające, które otrzymaliśmy za pomocą algorytmu Prima jest inne niż otrzymane za pomocą algorytmu Kruskala. Jednakże ich waga jest taka sama (obydwa są minimalne).

Algorytm Prima - uwagi

- Złożoność obliczeniowa algorytmu Prima to $O(|V(G)|^2)$.
- Pętle i krawędzie wielokrotne nie przeszkadzają w działaniu tego algorytmu.
- Jeśli graf jest niespójny i chcemy otrzymać na końcu spinający las (czyli po jednym drzewie spinającym na każdą składową), to musimy algorytm nieco zmodyfikować (ćwiczenie).