Wykłady z Matematyki Dyskretnej

dla kierunku Informatyka

dr Adam Marszałek

Instytut Informatyki
Politechnika Krakowska

Wykłady na bazie materiałów:
dra hab. Andrzeja Karafiata
dr hab. Joanny Kołodziej, prof. PK

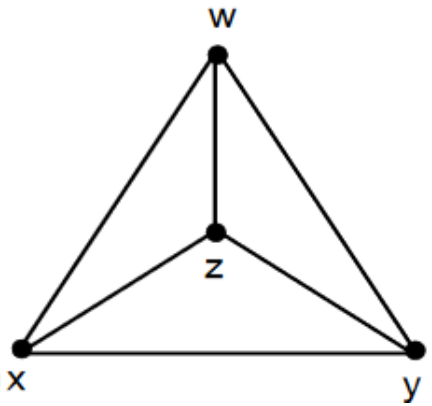
Droga i cykl Hamiltona

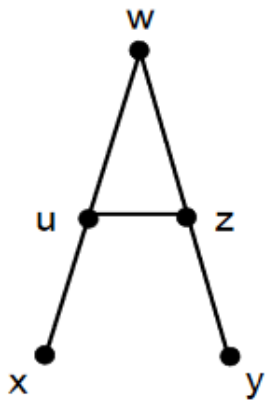
Definicja: Droga Hamiltona

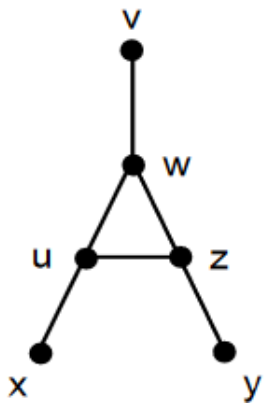
Drogą Hamiltona nazywamy drogę, która przechodzi przez każdy wierzchołek grafu dokładnie jeden raz. Graf posiadający drogę Hamiltona nazywamy grafem **półhamiltonowskim**.

Definicja: Cykl Hamiltona

Cyklem Hamiltona nazywamy cykl przechodzący przez wszystkie wierzchołki grafu. Graf posiadający cykl Hamiltona nazywamy **grafem hamiltonowskim**.







Kod Greya

Zacznijmy od następującej obserwacji:

Niech ciąg $(C_1, C_2, \ldots, C_m)$ zawiera wszystkie ciągi binarne C_i długości k (jest ich 2^k), przy czym C_i różni się od C_{i+1} na dokładniej jednej współrzędnej. Wówczas ciąg

$$C_1 0, C_2 0, \ldots, C_m 0, C_m 1, C_{m-1} 1, \ldots, C_1 1$$

zawiera wszystkie ciągi binarne długości $k+1$, przy czym każde dwa sąsiednie ciągi różnią się na dokładnie jednej współrzędnej.

W ten sposób konstruujemy prosty algorytm rekurencyjny generujący wszystkie ciągi binarne długości n.

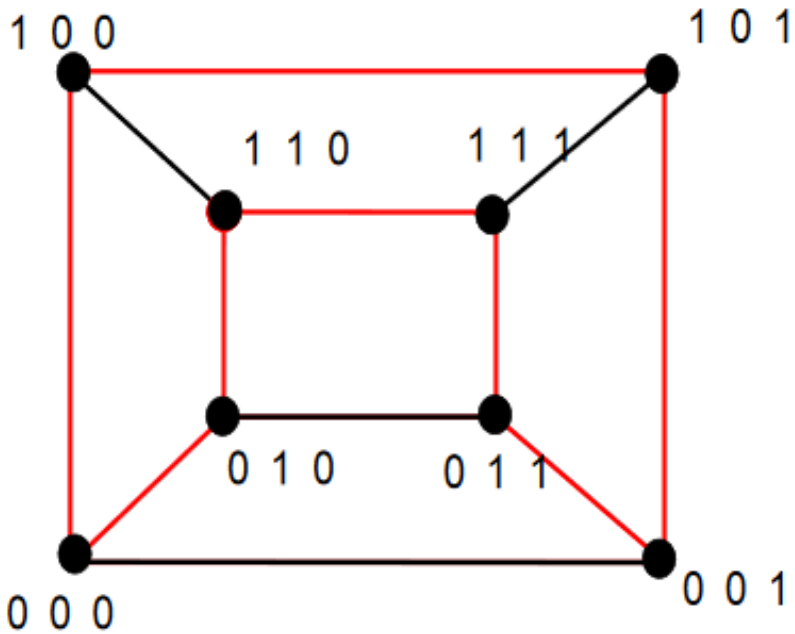
Otrzymany ciąg

$$C_1, C_2, \ldots, C_{2^n}$$

nazywamy **kodem binarnym Greya** rzędy n.

Cykl Hamiltona w kostce Q_n

W kostce n-wymiarowej Q_n, kolejność ciągów binarnych, wygenerowanych przez kod Greya stanowi drogę Hamiltona w tej kostce.



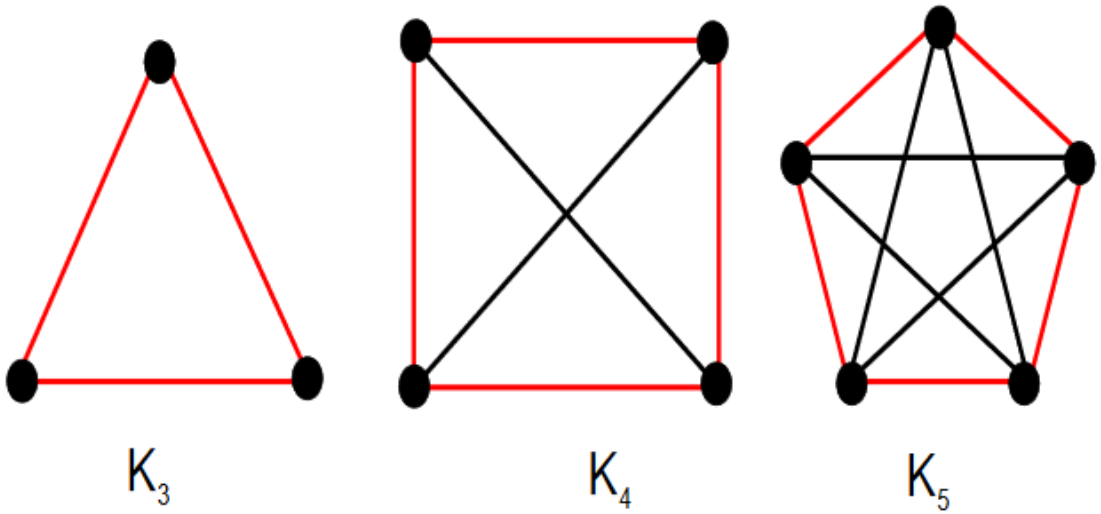
Warunki istnienia cyklu Hamiltona

Twierdzenie: Warunek konieczny

Jeżeli graf o n wierzchołkach jest hamiltonowski, to posiada co najmniej n krawędzi.

Twierdzenie

Każdy graf pełny K_n jest grafem Hamiltona.



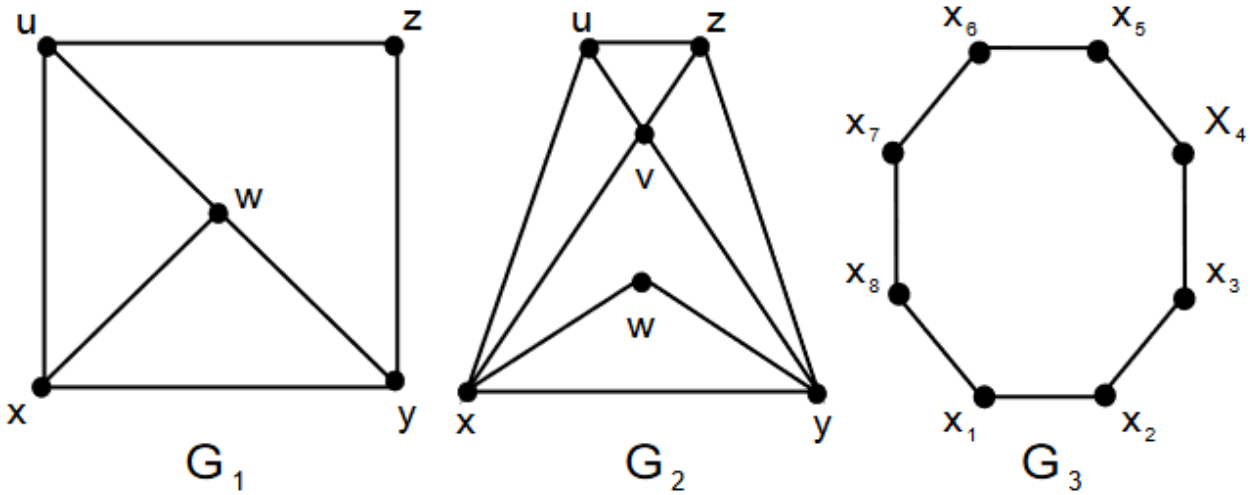
Warunki istnienia cyklu Hamiltona

Twierdzenie: Ore (1960)

Niech graf $G = \langle V, E \rangle$ będzie grafem spójnym i niech $|V| = n \geqslant 3$ (tzn. graf G ma co najmniej trzy wierzchołki). Jeżeli

$$deg(u) + deg(w) \geqslant n$$

dla każdej pary wierzchołków $u, w \in V$, które nie są połączone krawędzią, to graf G jest grafem hamiltonowskim.



Warunki istnienia cyklu Hamiltona

Twierdzenie: Dirac (1952)

Jeżeli w grafie prostym i spójnym $G = \langle V, E \rangle$ o n wierzchołkach ($n \geqslant 3$) oraz stopień każdego wierzchołka $u \in V$ spełnia warunek $deg(u) \geqslant \frac{1}{2}n$, to graf G jest grafem hamiltonowskim.

Twierdzenie

Jeżeli w grafie prostym i spójnym $G = \langle V, E \rangle$ o n wierzchołkach jest co najmniej $\frac{1}{2}(n-1)(n-2)+2$ krawędzi, to graf G jest grafem hamiltonowskim.

Uwaga

Graf hamiltonowski musi być spójny, więc wcześniejsze twierdzenia można traktować jako warunki wystarczające spójności grafu.

Cykl i droga Hamiltona w grafach dwudzielnych

Twierdzenie: Warunek konieczny

Niech $G = \langle V_1 \cup V_2, E \rangle$ będzie grafem dwudzielnym, wówczas

- jeśli G ma cykl Hamiltona, to $|V_1| = |V_2|$,
- jeśli G ma drogę Hamiltona, to $||V_1| - |V_2|| \leqslant 1$.

Twierdzenie: Warunek wystarczający

Niech $G = \langle V_1 \cup V_2, E \rangle$ będzie grafem pełnym dwudzielnym, wówczas

- jeśli $|V_1| = |V_2|$, to G ma cykl Hamiltona,
- jeśli $||V_1| - |V_2|| \leqslant 1$, to G ma drogę Hamiltona.

Cykle Hamiltona rozłączne krawędziowo

Definicja

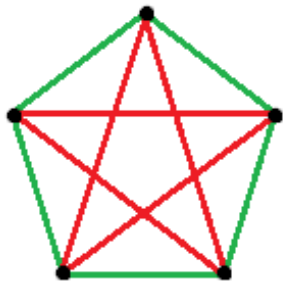
Dwa cykle są rozłączne krawędziowo, gdy każda krawędź należy tylko do jednego cyklu w grafie.

Twierdzenie

Graf pełny K_n zawiera $\left[\frac{n-1}{2}\right]$ rozłącznych krawędziowo cykli Hamiltona.

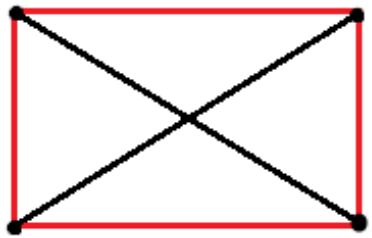
W grafie K_5 mamy

$$\left[\frac{5-1}{2}\right] = 2$$



W grafie K_4 mamy

$$\left[\frac{4-1}{2}\right] = \left[1\frac{1}{2}\right] = 1$$

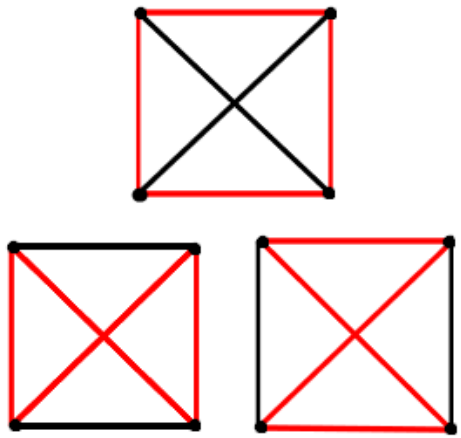


Cykle Hamiltona w grafie pełnym

Twierdzenie

Graf pełny K_n zawiera $\frac{(n-1)!}{2}$ różnych cykli Hamiltona.

W grafie K_4 mamy $\frac{(4-1)!}{2} = 3$ różne cykle Hamiltona:

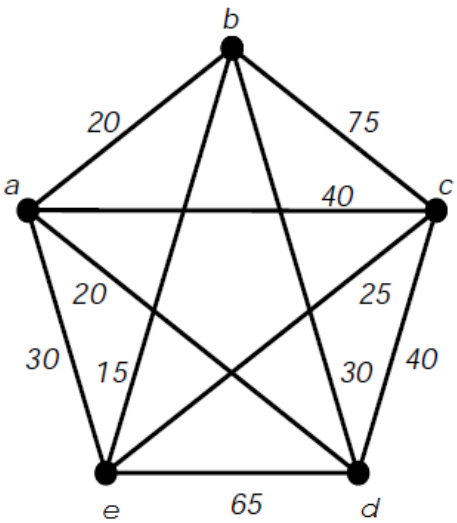


- W grafie K_5 mamy $\frac{(5-1)!}{2} = 12$ różnych cykli Hamiltona.
- W grafie K_20 mamy $\frac{(20-1)!}{2} > 10^{17}$ różnych cykli Hamiltona.

Problem komiwojażera

Komiwojażer ma odwiedzić kilka miast (każde dokładnie jeden raz) i powrócić do miasta, z którego wyruszył przebywając łącznie najkrótszą (najtańszą lub najszybciej przebytą) drogę. Znane są odległości (koszt lub czas) przejazdu między każdą parą miast. Należy wyznaczyć komiwojażerowi trasę przejazdu tak, aby mógł odwiedzić każde miasto dokładnie jeden raz i całkowita droga (koszt lub czas) podróży była/był możliwie najkrótsza/najmniejszy.

Problem ten możemy sformułować w teorii n-wierzchołkowej sieci pełnej, a następnie znaleźć najkrótszy (najtańszy, najszybszy) cykl Hamiltona o n wierzchołkach.



Przykładowo cykl a, b, c, d, e, a ma wagę 230, a cykl a, b, e, c, d, a ma wagę 110.

Problem komiwojażera

Teoretycznie problem komiwojażera można rozwiązać poprzez wyznaczenie $\frac{1}{2}(n-1)!$ cykli Hamiltona i wybranie tego, który ma najmniejszą sumę wag. Oczywiście, metoda ta jest bardzo nieefektywna. Bowiem, jeżeli dysponujemy komputerem sprawdzającym milion permutacji na sekundę, to:

- $n = 10$ liczba cykli wynosi $(10-1)!/2 = 181440$ czas obliczeń $= 1.8s$.
- $n = 20$ liczba cykli wynosi $(20-1)!/2 = 6082255020441600$ czas obliczeń ok. 29330 lat.

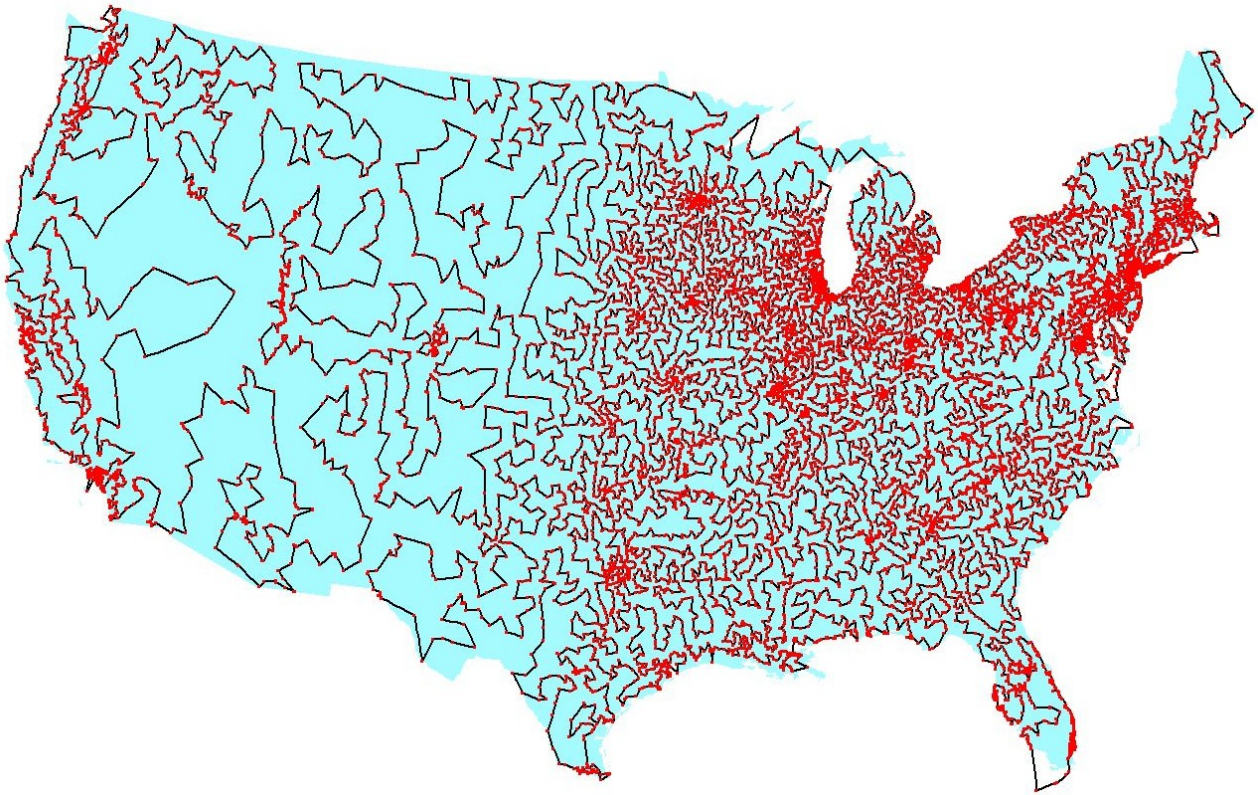
Problem komiwojażera jest NP-zupełny.

Rozwiązania problemu komiwojażera, możemy wybrać jedną z dwóch metod:

- metodą dokładną, np. metodą podziału i ograniczeń, która wygeneruje dokładne rozwiązanie, ale działającą w czasie wykładniczym (a więc metoda „wolna"),
- metodą przybliżoną (inaczej nazywaną metodą aproksymacyjną), która generuje rozwiązanie bliskie optymalnemu ale działającą w czasie wielomianowym.

Historia TSP - Travelling salesman problem

Big TSP problem:
http://www.tsp.gatech.edu//history/pictorial/dfj.html
N = 13509



Heurystyki przeszukiwań

Heurystyka (gr.heuriskein znaleźć, odkryć) to praktyczna, oparta na doświadczeniu reguła postępowania, która może znacznie uprościć lub skrócić proces rozwiązywania rozważanego problemu, gdy metoda rozwiązania nie jest znana lub jest zawiła i czasochłonna.

- Dla zadań, w których mamy do czynienia z wieloma rozwiązaniami, ważne jest wczesne odrzucenie nieobiecujących kierunków poszukiwania rozwiązania. Zapewnia to ogromne oszczędności na kosztach obliczeniowych, a w rezultacie przyspiesza znalezienie rozwiązania.
- Metody heurystyczne pozwalają na znalezienie w akceptowalnym czasie przynajmniej przybliżonego rozwiązania problemu, choć nie gwarantuje tego we wszystkich przypadkach.
- Skuteczności kroków heurystycznych nie można w pełni udowodnić teoretycznie, można jedynie pokazać doświadczalnie ich trafność.
- Algorytmy genetyczne i ewolucyjne, algorytmy mrówkowe itp.

Drzewo

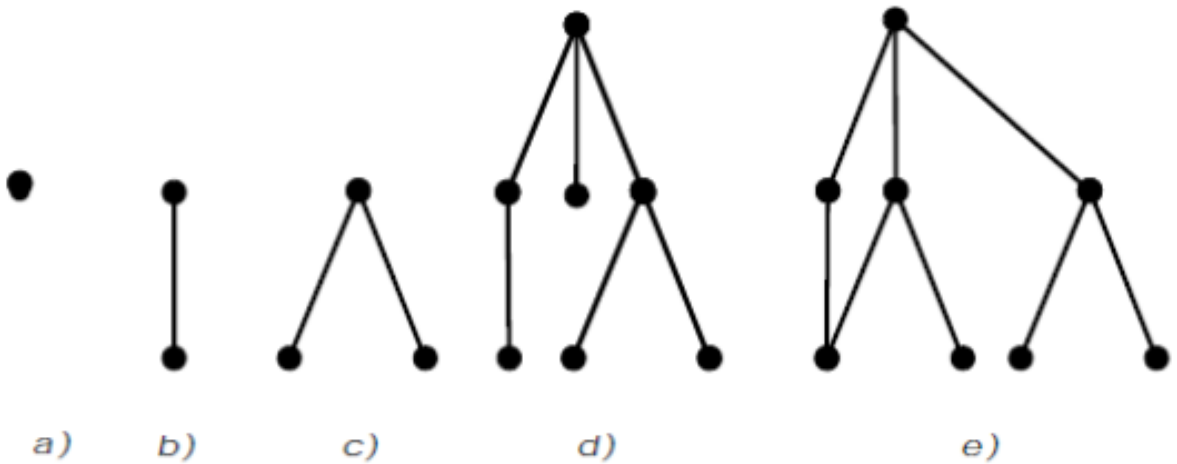
Definicja: Drzewo

Drzewem nazywamy graf (nieskierowany) spójny i acykliczny.

Z definicji wynika, że drzewo jest grafem prostym tzn. nie ma pętli i krawędzi wielokrotnych. W przeciwnym przypadku miałoby cykle, co przeczy definicji drzewa.

Definicja: Las

Graf niespójny i acykliczny nazywamy **lasem**. Jego składowe są drzewami.



Własności drzew

Twierdzenie

Niech T będzie grafem mającym n wierzchołków. Wtedy następujące warunki są równoważne:

1. T jest drzewem,
2. T jest acykliczny i ma $n-1$ krawędzi,
3. T jest spójny i ma $n-1$ krawędzi,
4. T jest spójny, ale przestaje być spójny po usunięciu dowolnej krawędzi,
5. każde dwa wierzchołki T połączone są dokładnie jedną drogą,
6. T jest grafem acyklicznym, ale po dodaniu dowolnej nowej krawędzi otrzymany graf ma dokładnie jeden cykl.

Własności drzew

Własność

Jeżeli graf G jest lasem, który ma n wierzchołków i k składowych, to graf G ma $n - k$ krawędzi.

Definicja: Liść

Wierzchołki drzewa mające stopień wierzchołka równe 1 nazywamy **liśćmi**.

Własność

Drzewo T mające co najmniej jedną krawędź ma co najmniej dwa liście.

Drzewa z wyróżnionym korzeniem

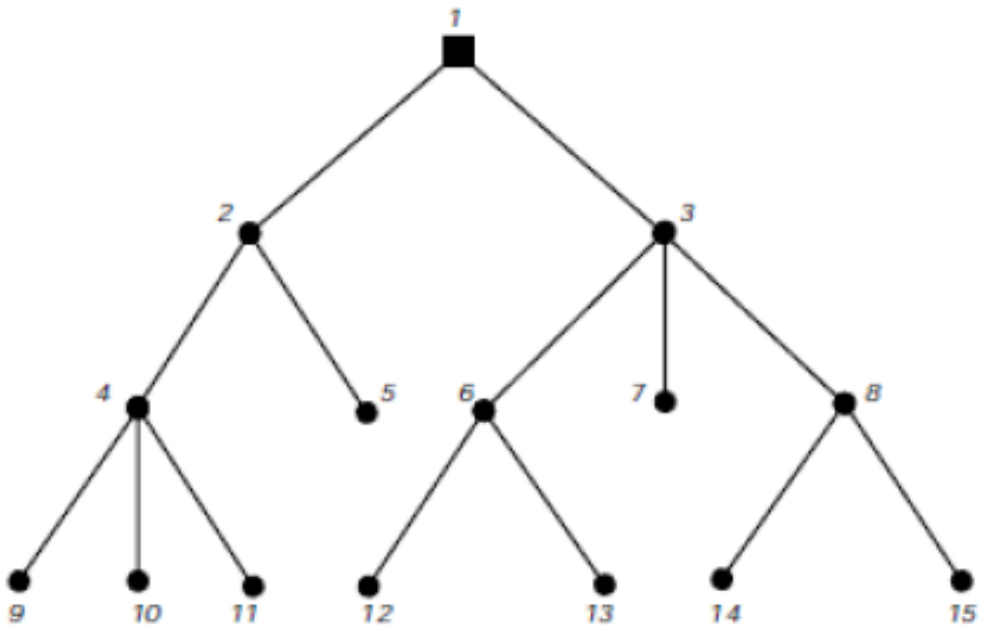
Definicja

W wielu przypadkach drzewa, które wykorzystujemy do rozwiązania problemów praktycznych mają strukturę hierarchiczną tzn. mają jeden wierzchołek wyróżniony zwany **korzeniem**. W takim drzewie liście nazywa się **węzłami zewnętrznymi** lub **końcowymi**. Pozostałe wierzchołki w drzewie nazywamy węzłami wewnętrznymi.

Definicja

Niech $\{v, w\}$ będzie krawędzią należącą do drzewa T. Wtedy jeżeli wierzchołek v jest bliżej korzenia, to v jest **rodzicem** w, a w jest **dzieckiem** v.

Drzewa z wyróżnionym korzeniem



- Każdy wierzchołek (poza korzeniem) ma dokładnie jednego rodzica.
- Rodzic może mieć kilkoro dzieci.
- Ogólnie w jest **potomkiem** v jeśli $w \neq v$ oraz wierzchołek v należy do drogi prostej z w do korzenia.

Drzewa z wagami

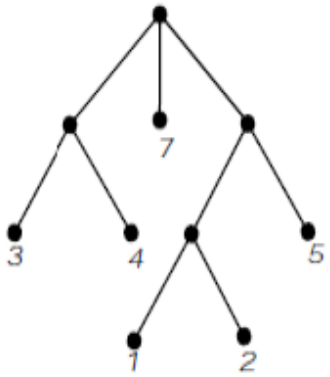
Definicja

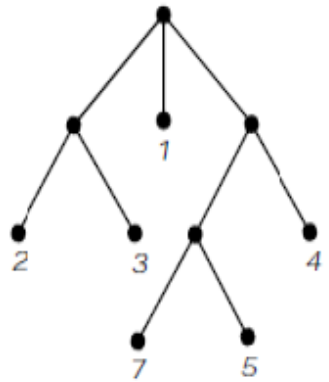
Drzewem z wagami nazywamy drzewo z korzeniem, w którym każdemu liściowi przyporządkowana jest liczba nieujemna, nazywaną wagą tego liścia.

Definicja

Niech drzewo T ma t liści, których wagami są liczby $w_1, w_2, \ldots, w_t$ oraz niech $l_1, l_2, \ldots, l_t$ oznaczają odpowiednio numery poziomów liści. Wówczas **wagą drzewa** T nazywamy liczbę $W(T) = \sum_{i=0}^{t} w_i l_i$

$W(T_1) = 40$





$W(T_2) = 55$

Drzewa optymalne z wagami

Definicja

Niech dany będzie zbiór wag $\{w_1, w_2, \ldots, w_n\}$ oraz drzewo T z korzeniem o n liściach. Drzewo T nazywamy **drzewem optymalnym z wagami** $\{w_1, w_2, \ldots, w_n\}$, jeżeli wagi te zostały tak przyporządkowane liściom, że $W(T)$ jest najmniejsza z możliwych.

Algorytm Huffmana:

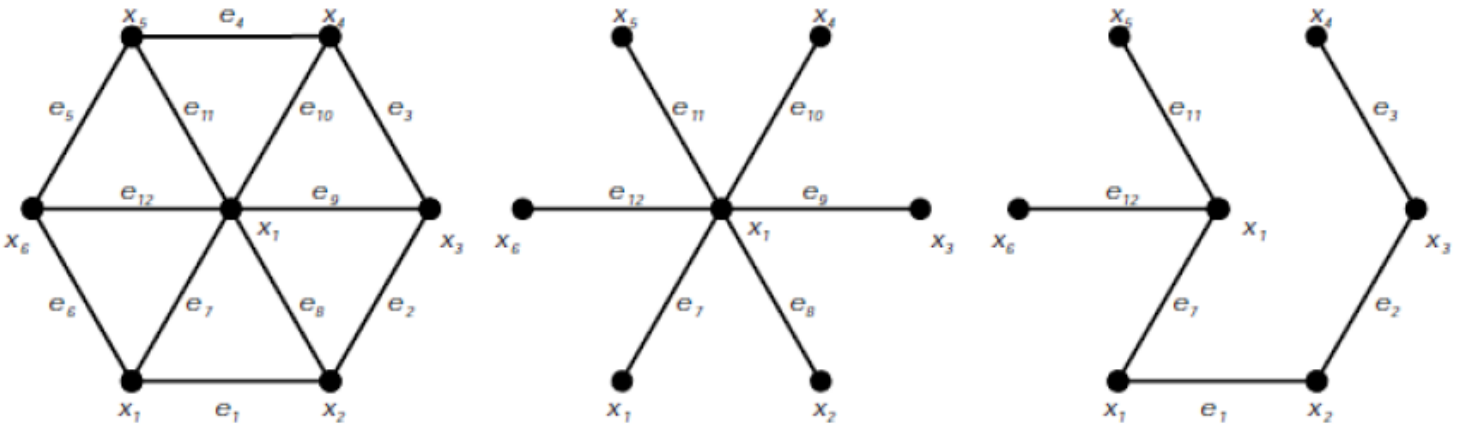
Dane: L lista t wierzchołków o wagach $\{w_1, ..., w_t\}$, $(t \geq 2)$
Wyniki: optymalne drzewo binarne $T(L)$

Huffman(L)
1 **if** $t = 2$
2 **then**
3 $T(L)$ drzewo z dwoma liśćmi o wagach w_1, w_2
4 **else**
5 z listy L wybierz wierzchołki o najmniejszych wagach u, v
6 połącz wybrane wierzchołki korzeniem o wadze $u + v$
7 z L usuń wybrane wierzchołki i dodaj wierzchołek (korzeń) o wadze $u + v$
8 Huffman(L)

Drzewa rozpinające grafu

Definicja: Drzewo rozpinające

Drzewo T nazywamy **drzewem rozpinającym (spinającym)** (lub **dendrytem**) spójnego grafu G, jeżeli jest podgrafem grafu $G(T \subset G)$ oraz zawiera wszystkie jego wierzchołki, czyli $V_G = V_T$.
Jeżeli graf G nie jest grafem spójnym, to zbiór drzew spinających każdej składowej tego grafu, nazywamy **lasem rozpinającym** grafu G.



Drzewa rozpinające grafu

Twierdzenie

Każdy graf spójny ma drzewo rozpinające.

Własność

Każde drzewo rozpinające T lub las drzew rozpinających otrzymujemy z grafu G przez usunięcie pewnej liczby krawędzi grafu G i pozostawienie wszystkich jego wierzchołków.

Definicja: Rząd cykliczności

Rzędem cykliczności lun **liczbą cyklometryczną** grafu G nazywamy liczbę

$$\gamma(G) = m - n + k,$$

gdzie m to liczba krawędzi, n - liczba wierzchołków, k liczba składowych w grafie G. Jest to minimalna liczba krawędzi potrzebnych do usunięcia aby graf G stał się drzewem (lasem).

Zliczanie drzew rozpinających

Twierdzenie: Cayley (1889)

Istnieje n^{n-2} różnych drzew oznakowanych mających n wierzchołków.

Wniosek

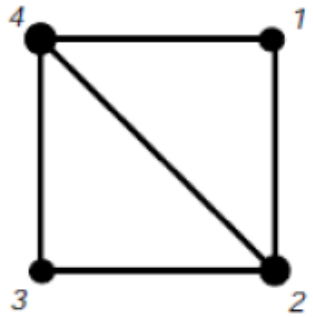
Graf K_n ma n^{n-2} drzew rozpinających.

Twierdzenie: Liczba drzew rozpinających

Niech G będzie spójnym grafem prostym o n wierzchołkach i niech $M = [m_{ij}]$ będzie macierzą wymiaru $n \times n$, taką że $m_{ij} = deg(v_i)$, $m_{ij} = -1$ gdy wierzchołki v_i i v_j są sąsiednie, oraz $m_{ij} = 0$ w przeciwnym razie. Wtedy liczba drzew spinających grafu G jest równa dopełnieniu algebraicznemu dowolnego wyrazu macierzy M.

Zliczanie drzew rozpinających

Rozważmy graf G



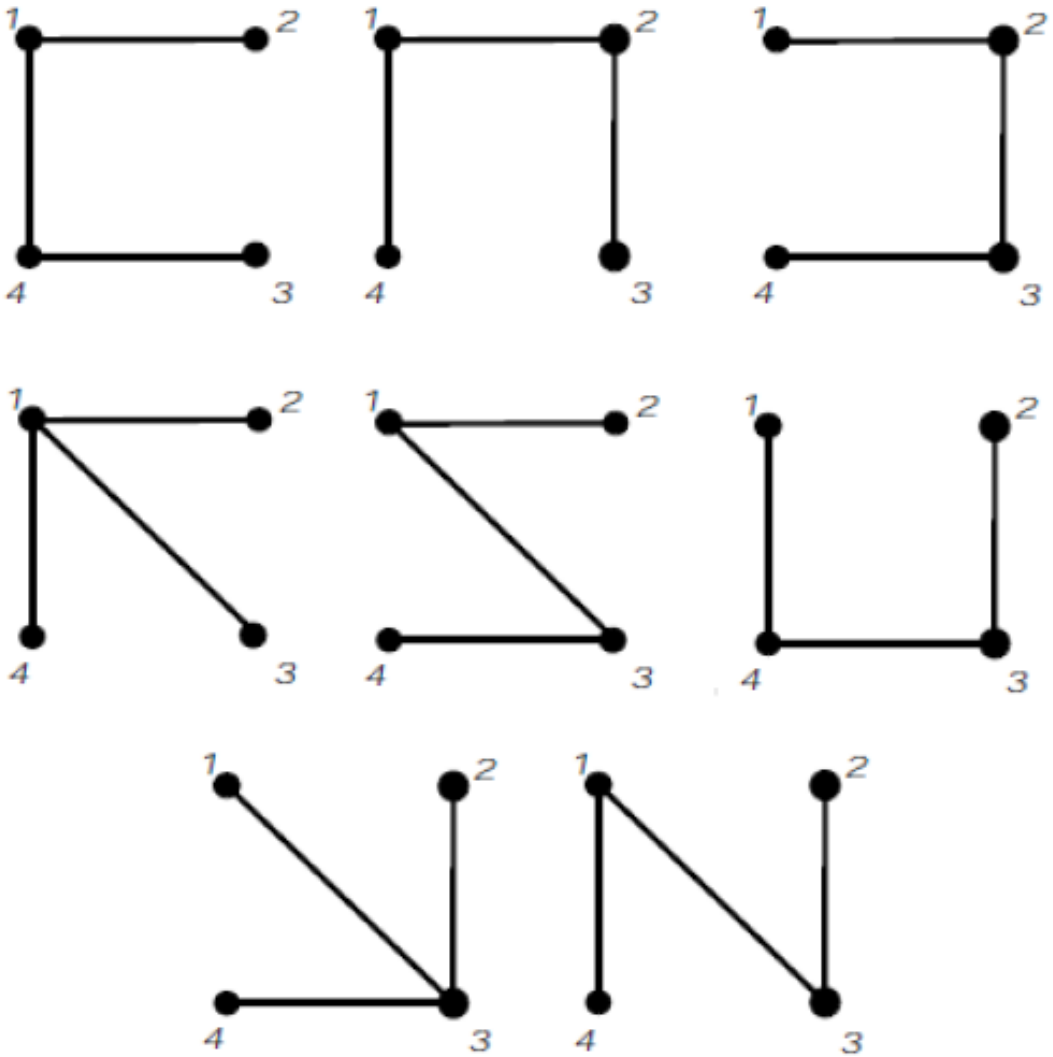
Macierz M ma postać:

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

Dopełnienie algebraiczne wyrazu m_{44} wynosi:

$$(-1)^{4+4} \det \left(\begin{bmatrix} 2 & -1 & 0 \\ -1 & 3 & -1 \\ 0 & -1 & 2 \end{bmatrix} \right) = 12 - (2+2) = 8$$

Drzewa rozpinające grafu

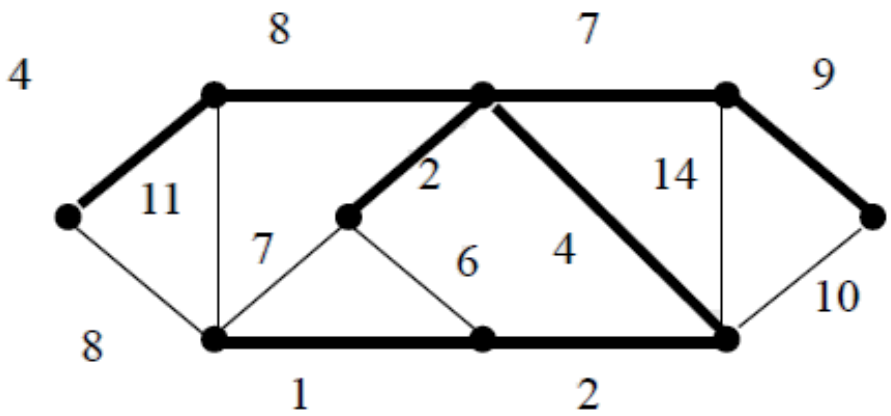


Minimalne drzewo rozpinające

Definicja

Dany jest graf spójny G z wagami. Drzewo rozpinające grafu G o najmniejszej wadze nazywamy **minimalnym drzewem spinającym (minimal spanning tree - MST)**.

- Waga krawędzi e - przypisana jej liczba nieujemna: $w(e)$.
- Waga grafu (podgrafu) G - suma wag jego krawędzi: $w(G) = \sum_{e \in E_G} w(e)$.



Algorytm Kruskala

Algorytm Kruskala generowania MST

Dane: Graf spójny G z wagami.

Szukane: Zbiór krawędzi minimalnego drzewa spinającego grafu G.

Algorytm:

Krok 1. Tworzymy pusty zbiór krawędzi T oraz listę L wszystkich krawędzi grafu G uporządkowanych według rosnących wag.

Krok 2. Dopóki w T nie ma wszystkich wierzchołków grafu G, wybieramy z listy L krawędź e_i (pierwszą jeszcz nie wybraną), jeśli nie tworzy ona cyklu z krawędziami już obecnymi w T to dodajemy ją do zbioru T.

Algorytm Kruskala

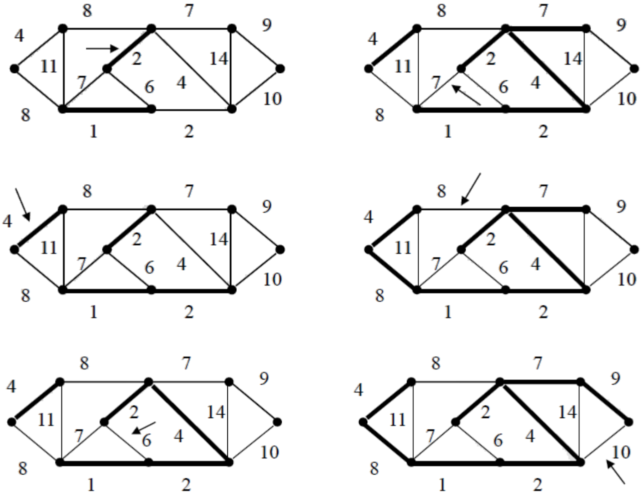
Twierdzenie

Algorytm Kruskala daje w wyniku minimalne drzewo spinające.

Uwagi

- Algorytm Kruskala działa poprawnie nawet gdy graf G ma pętle i krawędzie wielokrotne. Pętle pomija, a z krawędzi wielokrotnych wybiera tylko tą o najmniejszej wadze.
- Algorytm Kruskala zastosowany do grafu niespójnego daje w wyniku las spinający złożony z minimalnych drzew spinających poszczególnych składowych grafu.

Algorytm Kruskala: Przykład



Algorytm Prima

Algorytm Prima generowania MST

Dane: Graf spójny G z wagami.

Szukane: Zbiór krawędzi minimalnego drzewa spinającego grafu G.

Algorytm:

Krok 1. Wybieramy w grafie dowolny wierzchołek startowy.

Krok 2. Dopóki drzewo T nie ma wszystkich wierzchołków grafu G, znajdujemy krawędź o najniższej wadze spośród wszystkich krawędzi prowadzących od wybranych już wcześniej wierzchołków do wierzchołków jeszcze nie wybranych. Znalezioną krawędź dodajemy do drzewa spinającego T.

Algorytm Prima

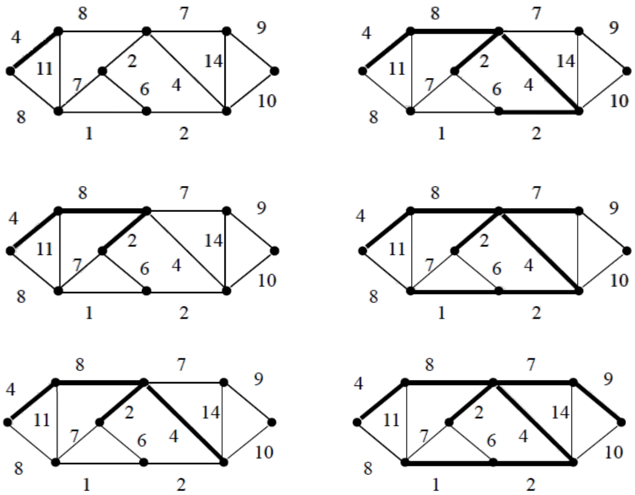
Twierdzenie

Algorytm Prima daje w wyniku minimalne drzewo spinające.

Uwagi

- Algorytm Kruskala i Prima wybierają krawędzie w różnej kolejności, lecz wynik końcowy jest taki sam (waga minimalnego drzewa spinającego, niekoniecznie samo drzewo).
- W algorytm Prima kolejność wyboru krawędzi zależy od wyboru wierzchołka startowego.

Algorytm Prima: Przykład



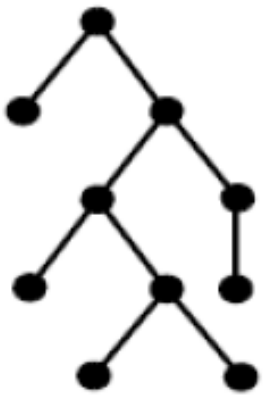
Drzewa binarne

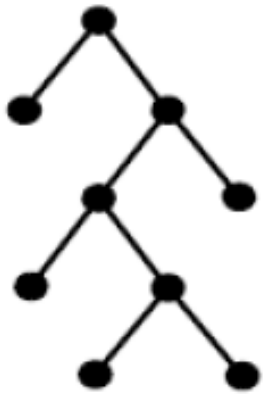
Definicja: Drzewo binarne

Drzewem binarnym nazywamy drzewo z wyróżnionym korzeniem, które posiada wierzchołki stopnia co najwyżej trzeciego (w którym każdy węzeł ma co najwyżej dwóch synów (lewy syn, prawy syn) lub w ogóle nie ma synów).

Definicja: Regularne drzewo binarne

Regularnym drzewem binarnym nazywamy nie puste drzewo binarne, którego każdy węzeł ma dokładnie dwóch synów.





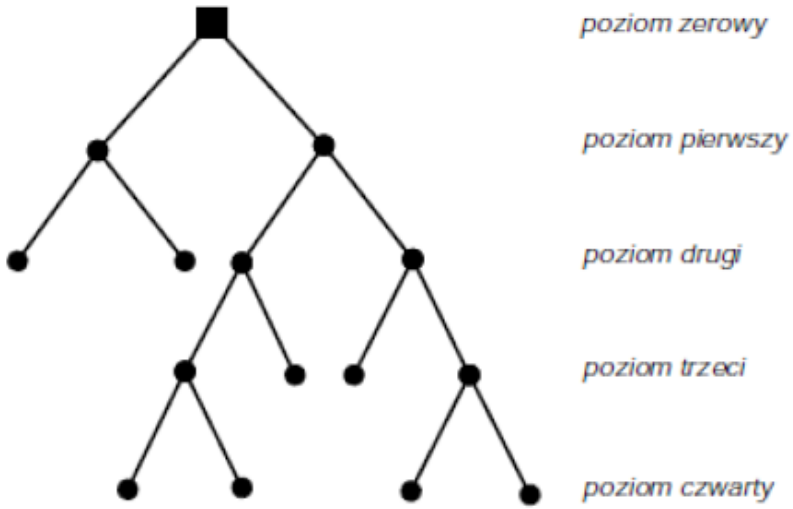
Drzewa binarne

Definicja: Poziom wierzchołka

Mówimy, że wierzchołek v w drzewie binarnym jest na poziomie l, jeżeli v jest w odległości l od korzenia (zakładamy, że korzeń jest na poziomie 0). Jest to długość jedynej drogi prostej z korzenia do wierzchołka v.

Definicja: Wysokość drzewa

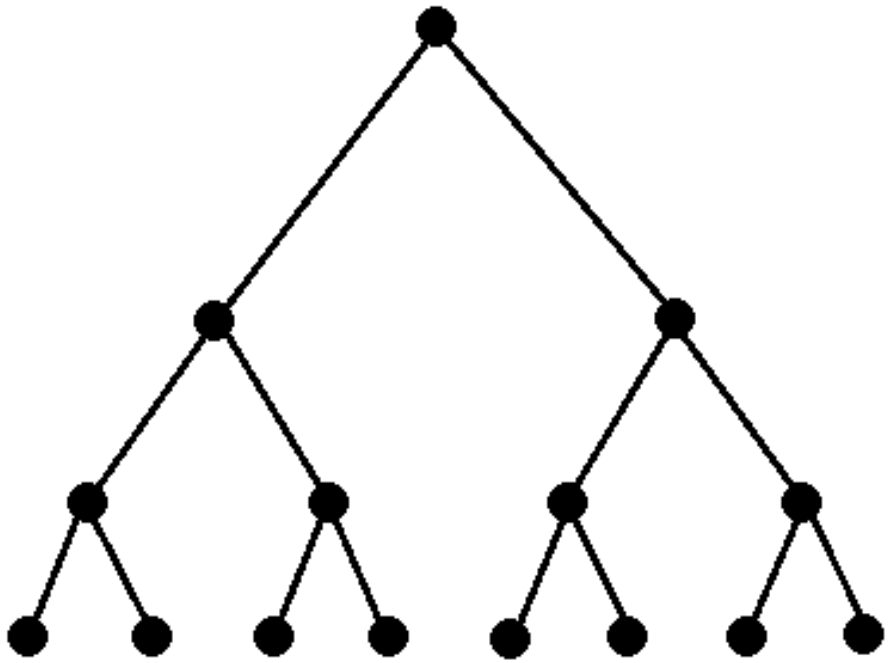
Maksymalny poziom l_{max} wierzchołka w drzewie binarnym nazywamy **wysokością drzewa**.



Drzewa binarne

Definicja: Pełne drzewo binarne

Pełnym drzewem binarnym nazywamy regularne drzewo binarne, w którym wszystkie liście mają ten sam numer poziomu, równy wysokości drzewa.



Drzewa binarne

Twierdzenie

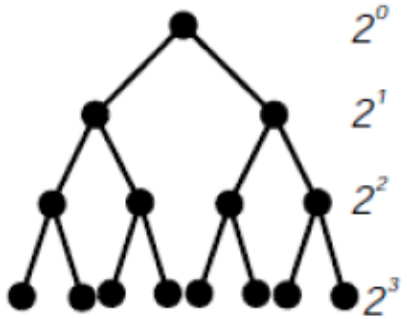
Liczba wierzchołków w regularnym drzewie binarnym jest zawsze nieparzysta.

Twierdzenie

Liczba liści w regularnym drzewie binarnym o n wierzchołkach wynosi $\frac{n+1}{2}$.

Twierdzenie

Maksymalna liczba wierzchołków w drzewie binarnym o k poziomach wynosi $n_{max} = 2^0 + 2^1 + \ldots + 2^k = 2^{k+1} - 1$.



$$n_{max} = 2^0 + 2^1 + 2^2 + 2^3 = 1 + 2 + 4 + 8 = 15$$

Przeszukiwanie drzewa binarnego w głąb

Algorytm przeszukiwania drzewa jest to algorytm, który wypisuje (odwiedza) wszystkie wierzchołki skończonego, uporządkowanego drzewa z wyróżnionym korzeniem. Istnieją trzy najbardziej popularne algorytmy, według których realizowane jest odwiedzanie kolejnych poddrzew i w konsekwencji węzłów drzewa. Są to:

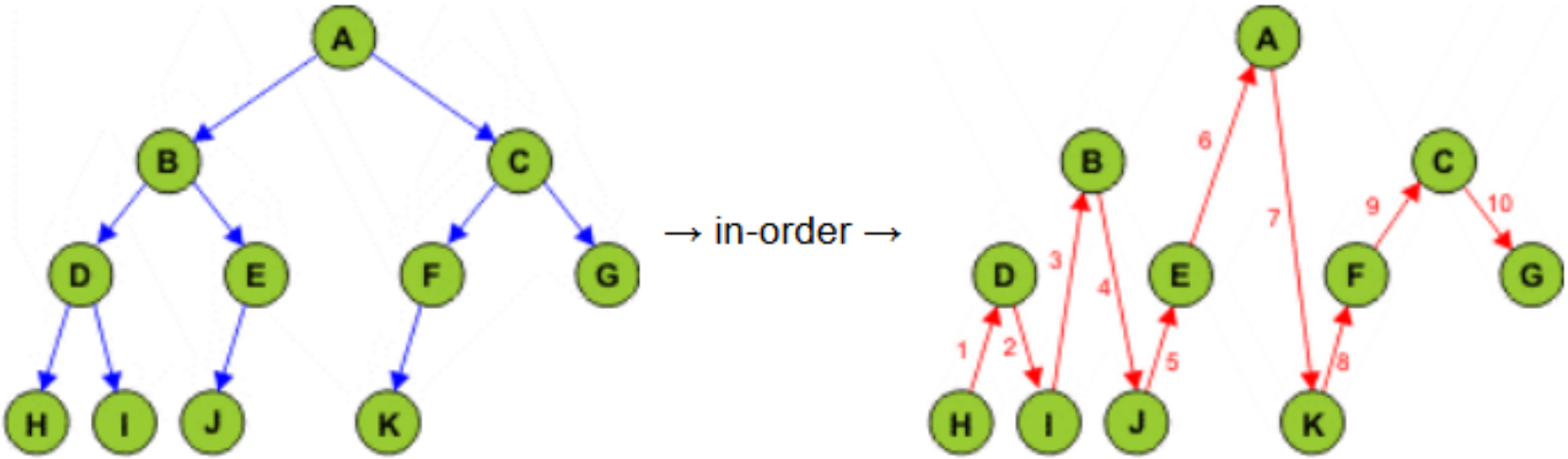
- przechodzenie w porządku **INORDER**, nazywane inaczej porządkiem wrostkowym,
- przechodzenie w porządku **PREORDER**, nazywane inaczej porządkiem przedrostkowym,
- przechodzenie w porządku **POSTORDER**, nazywany inaczej porządkiem przyrostkowym.

We wszystkich tych algorytmach zakłada się , że najpierw odwiedzane jest lewe poddrzewo korzenia, a następnie prawe.

Algorytm przeszukiwania INORDER

Algorytm INORDER (przejście poprzeczne, (T_L, λ, T_p))

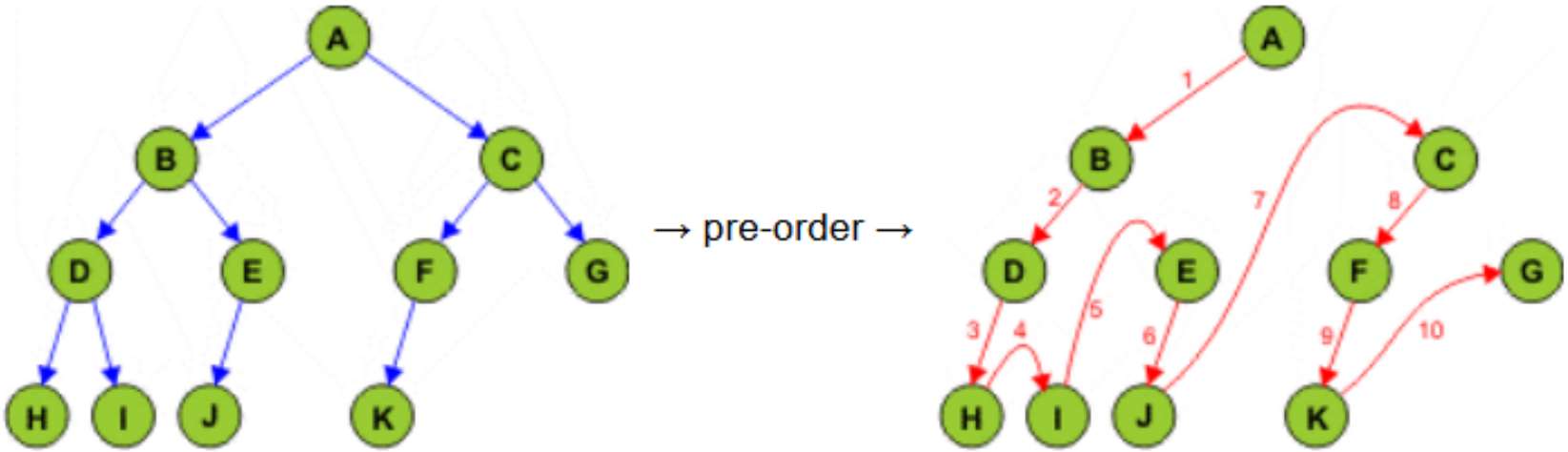
1. odwiedź poddrzewo lewe,
2. odwiedź korzeń,
3. odwiedź poddrzewo prawe.



Algorytm przeszukiwania PREORDER

Algorytm PREORDER (przejście wzdłużne, (λ, T_L, T_p))

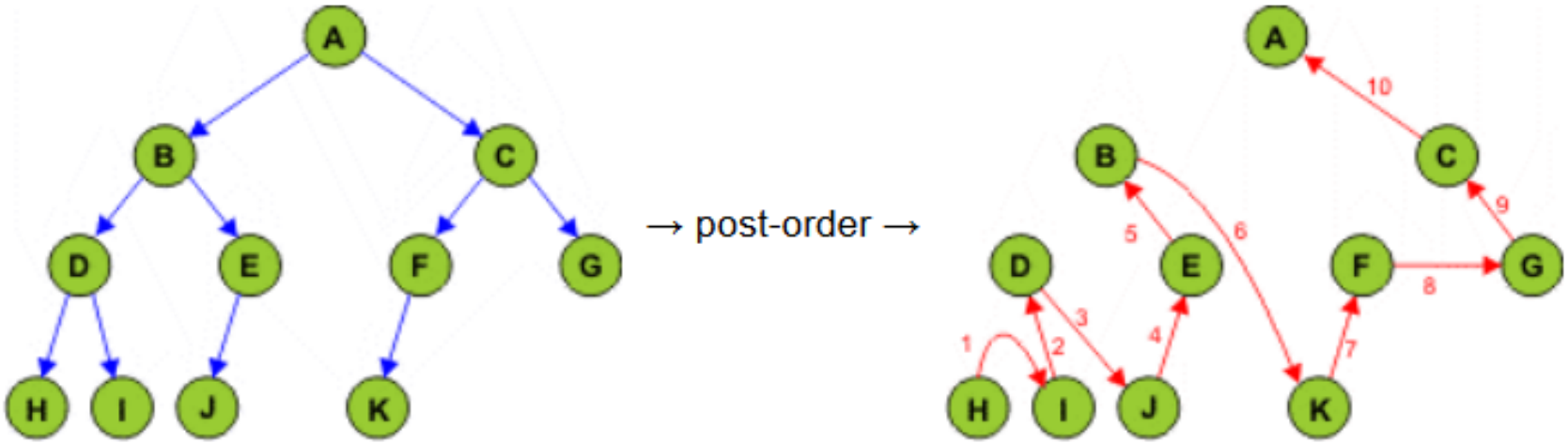
1. odwiedź korzeń,
2. odwiedź poddrzewo lewe,
3. odwiedź poddrzewo prawe.



Algorytm przeszukiwania POSTORDER

Algorytm PREORDER (przejście wsteczne, (T_L, T_p, λ))

1. odwiedź poddrzewo lewe,
2. odwiedź poddrzewo prawe,
3. odwiedź korzeń.



Przeszukiwanie drzewa z użyciem stosu lub kolejki

- **Lista** - uporządkowany ciąg elementów.
- **Koleja** - lista z trzema operacjami:
  - Dodawanie nowego elementu na koniec kolejki.
  - Zdejmowanie pierwszego elementu z początku kolejki.
  - Sprawdzanie, czy kolejka jest pusta.

  Taki sposób dodawania i odejmowanie elementów jest określany FIFO (ang. First in first out, pierwszy, który wszedł pierwszy wyjdzie).
- **Stos** - lista z trzema operacjami:
  - Dodawanie elementu na wierzch stosu.
  - Zdejmowanie elementu z wierzchu stosu.
  - Sprawdzanie, czy stos jest pusty.

  Sposób ten określany jest jako LIFO (ang. Last in first out, ostatni, który wszedł, pierwszy wyjdzie).

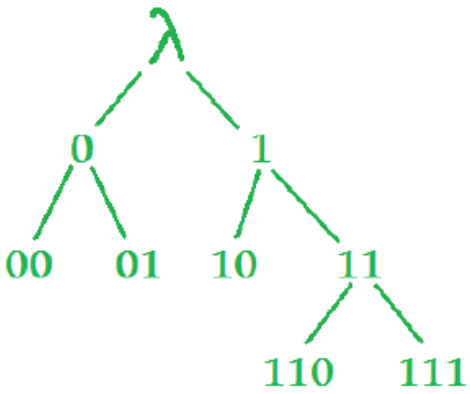
Przeszukiwanie drzewa w głąb (z użyciem stosu)

Przeszukiwanie drzewa w głąb z użyciem stosu:

1. Odwiedzamy korzeń λ i wkładamy go na stos.
2. Dopóki stos nie jest pusty powtarzamy (v- wierzchołek na wierzchu stosu)
   - Sprawdzamy, czy istnieje syn u wierzchołka v, który nie był odwiedzony:
     - Jeśli takie u znajdziemy, to odwiedzamy u i wkładamy go na wierzch stosu.
     - Jeśli takie u nie znajdziemy, to zdejmujemy v ze stosu.

Dodatkowo algorytm ten powinien zapamiętywać, jaki wierzchołek był zdjęty ze stosu. Ułatwia to stwierdzenie, który z synów wierzchołka znajdującego się na stosie pod spodem należy rozpatrzyć.

Przykład



Wierzchołek	Stos	Wierzchołek	Stos
λ	λ	10	λ,1,10
0	λ,0	1	λ,1
00	λ,0,00	11	λ,1,11
0	λ,0	110	λ,1,11,110
01	λ,0,01	11	λ,1,11
0	λ,0	111	λ,1,11,111
λ	λ	11	λ,1,11
1	λ,1	1	λ,1

Tabela pokazuje, jaki wierzchołek jest odwiedzany i jaka jest zawartość stosu po każdej iteracji.
W metodzie tej po każdym kroku algorytmu wierzchołki znajdujące się na stosie tworzą ścieżkę od wierzchołka wejściowego do wierzchołka aktualnie odwiedzanego.

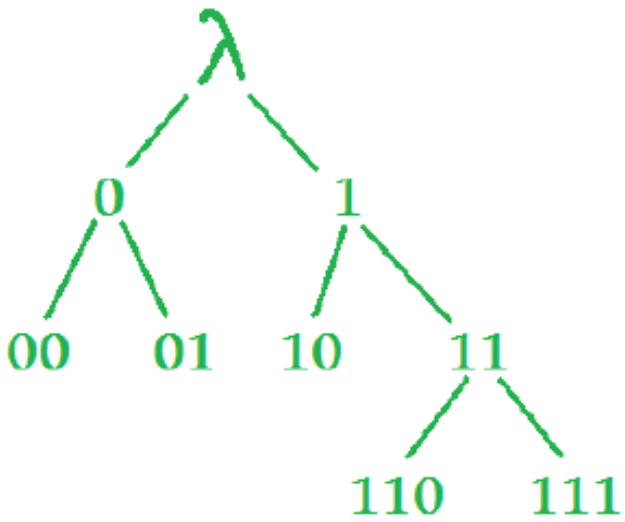
Przeszukiwanie drzewa wszerz (z użyciem kolejki)

Przeszukiwanie drzewa w głąb z użyciem kolejki:

1. Odwiedzamy korzeń λ i wkładamy go do kolejki.
2. Dopóki kolejka nie jest pusta, powtarzamy:
   - Bierzemy jeden wierzchołek v z początku kolejki.
   - Odwiedzamy wszystkich synów wierzchołka v.
   - Każdego wkładamy na koniec kolejki.

W metodzie przeszukiwania wszerz wierzchołki są przeszukiwane w kolejności od wierzchołków będących najbliżej wierzchołka początkowego do wierzchołków będących dalej

Przykład



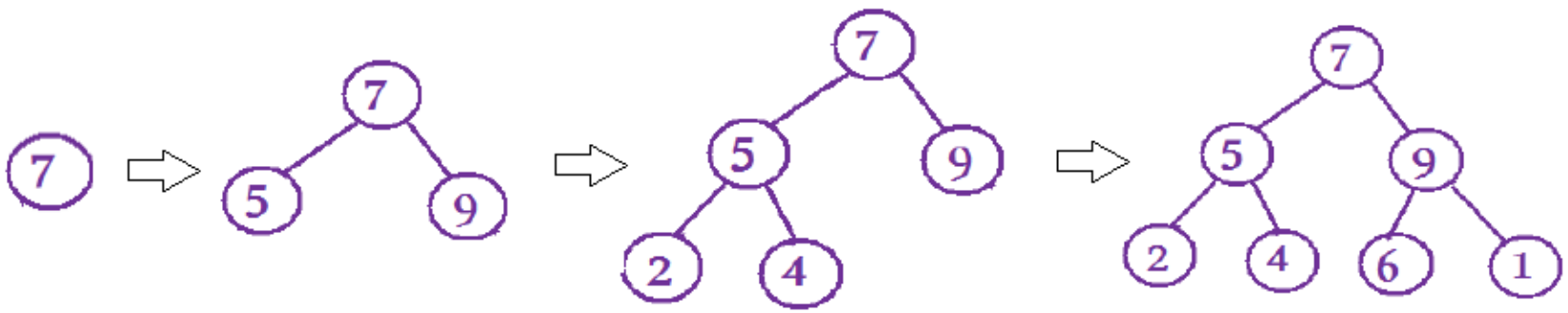
Wierzchołki	Kolejka
λ	λ
0,1	0,1
00,01	1,00,01
10,11	00,01,10,11
-	01,10,11
-	10,11
-	11
110,111	110,111
-	111
-	-

Tabela ukazuje odpowiednie wierzchołki i zawartości kolejki po każdej iteracji algorytmu dla drzewa z rysunku.

Przykład

Skonstruuj drzewo binarne z elementów zbioru $\{7, 5, 9, 2, 4, 6, 1\}$

1. Konstrukcję drzewa binarnego rozpoczynamy od korzenia, który jest pierwszym elementem zbioru, czyli liczbą 7.
2. Do korzenia dołączamy dwa węzły potomne, które leżą obok w zbiorze. Są to dwa kolejne elementy, 5 i 9.
3. Do lewego węzła potomnego 5 dołączamy jego węzły potomne. Są to kolejne liczby w zbiorze, czyli 2 i 4.
4. Pozostaje nam dołączyć do prawego węzła ostatnie dwa elementy zbioru, czyli liczby 6 i 1. Drzewo jest kompletne.



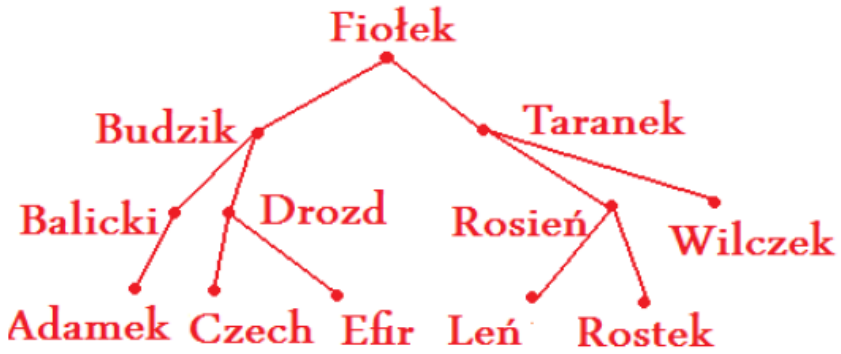
Drzewo BST

Drzewo BST

Ciąg liczb lub plików może być zorganizowany w specjalny rodzaj drzewa z wyróżnionym korzeniem, jest to **drzewo poszukiwań binarnych - BST**. Jest to drzewo binarne, w którym każdy węzeł ma przyporządkowaną jedną wartość, przy czym: wartość przechowywana w węźle jest większa od wartości przechowywanej w lewym synu i mniejsza od wartości przechowywanej w prawym synu.

- Przechodząc drzewo BST metodą INORDER uzyskuje się ciąg wartości posortowanych rosnąco.
- Aby znaleźć węzeł drzewa, w którym jest wartość x, porównujemy x z wartością k w korzeniu i w zależności od wyniku porównania:
  - $x = k$ – kończymy poszukiwania,
  - $x < k$ – szukamy w lewym poddrzewie,
  - $x > k$ – szukamy w prawym poddrzewie.

Przykład



W celu znalezienia rekordu klienta **Rostek** postępujemy następująco:

- Porównujemy nazwisko Rostek z nazwiskiem z korzenia, Fiołek. Rostek w porządku alfabetycznym znajduje się później, więc wybieramy prawą gałąź.
- Szukane nazwisko porównujemy z nazwiskiem Taranek. Rostek jest wcześniej w porządku alfabetycznym, wybieramy gałąź lewą.
- Porównujemy nazwisko Rostek z Rosieniem. Szukane nazwisko występuje później w porządku alfabetycznym wybieramy prawą gałąź.
- Dotarliśmy do szukanego wierzchołka, kończymy poszukiwania.

Opisana procedura jest skuteczna, ponieważ z każdego wierzchołka wychodzą co najwyżej dwie krawędzie w dół, więc wystarczy tylko kilka porównań by znaleźć właściwy adres, nawet jeśli liczba rekordów jest duża.

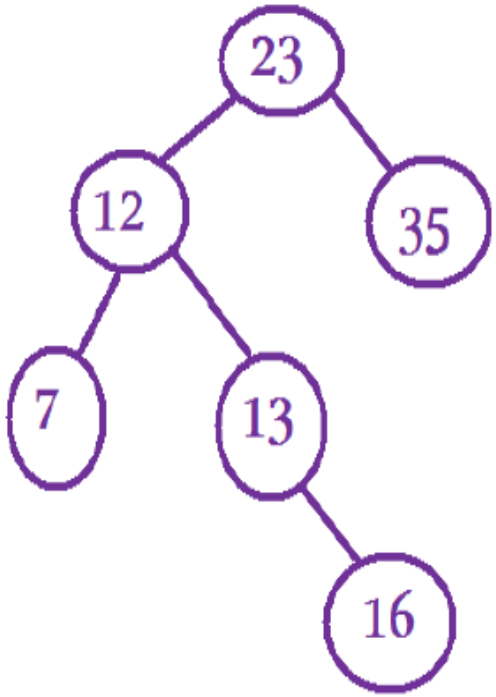
Przykład

Skonstruuj drzewo binarne BST dla ciągu wartości $\{23, 12, 35, 13, 16, 7\}$:

1. Konstrukcję drzewa binarnego rozpoczynamy od korzenia, który jest pierwszym elementem zbioru, czyli liczbą 23.
2. Do korzenia dołączamy dwa węzły potomne, które leżą obok w zbiorze. Są to dwa kolejne elementy, 12 i 35.Ich rozmieszczenie nie jest przypadkowe, mniejszą wartość-12 umieszczamy po lewej stronie, natomiast 35 po stronie prawej.
3. Kolejną liczbą z listy jest 13. Wartość ta jest mniejsza od 23 i większa od 12 dlatego zostaje prawym potomkiem 12.
4. Liczba 16 jest mniejsza od 23, większa od 12 i od 13 dlatego umieszczamy ją na pozycji potomka prawego 13.
5. Pozostała liczba 7 zostaje lewym potomkiem liczby 12, jako że jest mniejsza od 23 i 12.

Przykład

Skonstruuj drzewo binarne BST dla ciągu wartości $\{23, 12, 35, 13, 16, 7\}$:

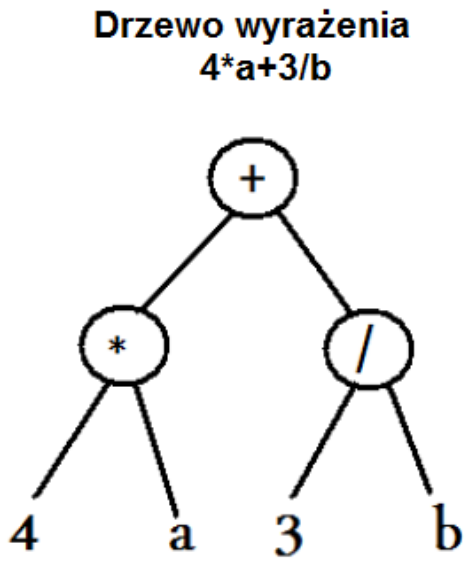


Drzewo wyrażeń algebraicznych

Drzewo wyrażeń arytmetycznych:

- Jest to przykład zastosowania drzew binarnych.
- W drzewie tym każdy wierzchołek ma etykietę.
- Liście etykietowane są stałymi bądź zmiennymi.
- Wierzchołki niebędące liśćmi etykietowane są operacjami arytmetycznymi.
- Każdemu wierzchołkowi w drzewie można przypisać wyrażenie arytmetyczne według zasady:
  - Dla liści wyrażeniami są etykiety tych liści(stałe lub zmienne)
  - Jeżeli wierzchołek v ma etykietę $\oplus$, a jego synom przypisano wyrażenie $W(v_0)$ i $W(v_1)$, to wierzchołkowi v przypisujemy wyrażenie $W(v) = (W(v_0) \oplus W(v_1))$.

Drzewo wyrażeń algebraicznych



- INORDER: $4 * a + 3/b$
- PREORDER: $+ * 4a/3b$ (Notacja Polska)
- POSTORDER: $4a * 3b/+$ (Odwrotna Notacja Polska)

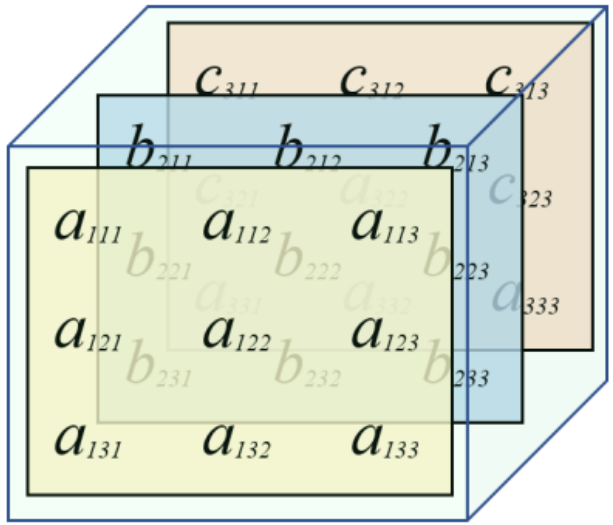
Zapis w notacji polskiej i odwrotnej polskiej nie wymaga nawiasów.

Drzewo wyrażeń algebraicznych

TensorFlow

Google TensorFlow

- Originally developed by the **Google** Brain Team within Google's Machine Intelligence research organization
- TensorFlow provides primitives for defining functions on **tensors** and automatically computing their derivatives.
- An open source software library for numerical computation using **data flow graphs**



Drzewo wyrażeń algebraicznych

Data Flow Graph

- Computations are represented as **graphs**:
  - Nodes are the operations (*ops*)
  - Edges are the *Tensors* (multidimensional arrays)
- Typical program consists of 2 phases:
  - **construction phase**: assembling a graph (model)
  - **execution phase**: pushing data through the graph