

Wojciech Kordecki
Anna Łyczkowska-Hanćkowiak

MATEMATYKA DYSKRETNA DLA INFORMATYKÓW



Helion 

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Redaktor prowadzący: Małgorzata Kulik

Projekt okładki: Studio Gravite / Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki

Grafika na okładce została wykorzystana za zgodą Shutterstock.com

Wydawnictwo HELION
ul. Kościuszki 1c, 44-100 GLIWICE
tel. 32 231 22 19, 32 230 98 63
e-mail: helion@helion.pl
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
<http://helion.pl/user/opinie/madyin>
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-283-4011-4

Copyright © Helion 2018

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to!» Nasza społeczność](#)

Spis treści

Oznaczenia	1
Wstęp	3
I. Podstawy	7
1. Indukcja i rekurencja	9
1.1. Zasada indukcji matematycznej	9
1.2. Zbiory częściowo uporządkowane	11
1.3. Rekurencja	15
1.4. Zadania	23
2. Rozmieszczenia i permutacje	27
2.1. Funkcje i rozmieszczenia	27
2.2. Permutacje	30
2.3. Liczby Stirlinga pierwszego rodzaju	38
2.4. Zadania	41
3. Kombinacje	44
3.1. Współczynnik dwumianowy	44
3.2. Generowanie podzbiorów	48
3.3. Zbiory z powtórzeniami	50
3.4. Zadania	52

4. Podziały	54
4.1. Podziały zbioru	54
4.2. Zasada szufladkowa Dirichleta	55
4.3. Zasada włączania-wyłączania	58
4.4. Liczby Stirlinga drugiego rodzaju	63
4.5. Podziały liczb	67
4.6. Zadania	69
5. Funkcje tworzące	72
5.1. Szeregi formalne	72
5.2. Rozwiązywanie rekurencji	75
5.3. Zastosowania funkcji tworzących	77
5.4. Sploty	79
5.5. Zadania	82
II. Grafy	83
6. Elementy teorii grafów	85
6.1. Podstawowe pojęcia	85
6.2. Macierze grafów	91
6.3. Izomorfizm, podstawowe własności i typy grafów . . .	102
6.4. Kolorowanie i wielomiany	111
6.5. Zadania	115
7. Cykle, drzewa, pokrycia	120
7.1. Grafy Eulera i Hamiltona, turnieje	120
7.2. Spójność	127
7.3. Drzewa	129
7.4. Skojarzenia	140
7.5. Zadania	144

8. Wybrane algorytmy grafowe	148
8.1. Algorytmy przeszukiwania	148
8.2. Minimalne drzewa spinające i minimalne drogi	152
8.3. Przepływy w sieciach	157
8.4. Zadania	164
III. Ogólne struktury kombinatoryczne	167
9. Ciała i przestrzenie wektorowe	169
9.1. Ciała skończone	169
9.2. Skończone przestrzenie wektorowe	171
9.3. Skończone geometrie rzutowe i afiniczne	175
9.4. Zadania	179
10. Matroidy	181
10.1. Podstawy	181
10.2. Transwersale	189
10.3. Matroidy dualne	192
10.4. Wielomiany Tutte'a	196
10.5. Zadania	200
11. Systemy i algorytmy zachłanne	203
11.1. Systemy zachłanne	203
11.2. Algorytmy zachłanne	207
11.3. Zadania	212
Rozwiązania, odpowiedzi i wskazówki	215
Rozdział 1	217
Rozdział 2	223
Rozdział 3	227

Rozdział 4	230
Rozdział 5	233
Rozdział 6	236
Rozdział 7	241
Rozdział 8	246
Rozdział 9	250
Rozdział 10	253
Rozdział 11	256
Dodatki	259
A. Podstawowe pojęcia	261
A.1. Notacja	261
A.2. Zbiory	262
A.3. Algebra	263
A.4. Rozwinięcie funkcji w szereg potęgowy	265
A.5. Prawdopodobieństwo	266
B. Sławni matematycy	267
Literatura	283
Skorowidz	285

8.2. Minimalne drzewa spinające i minimalne drogi

Niech $G = (V, E)$ będzie grafem nieskierowanym bez pętli o n wierzchołkach. Na zbiorze krawędzi E określamy funkcję o wartościach nieujemnych, $w : E \rightarrow \mathbb{R}^+$. Wartość $w(e)$ nazywamy wagą krawędzi e . Jeśli $e = \{u, v\}$, to piszemy po prostu $w(u, v)$. Graf G z określoną funkcją w nazywa się grafem ważonym.

Minimalne drzewo spinające

Minimalnym drzewem spinającym spójnego grafu G nazywamy drzewo spinające T takie, że jego waga

$$w(T) = \sum_{e \in T} w(e) \quad (8.2.1)$$

jest najmniejsza. Gdy G nie jest spójny, to analogicznie minimalnym lasem spinającym T grafu G nazywamy las spinający T_G taki, że waga $w(T_G)$ określona wzorem (8.2.1) jest najmniejsza.

Droga minimalna

Minimalną drogą łączącą wierzchołki u oraz v w grafie G , $u \neq v$, nazywamy drogę $P(u, v)$ taką, że jej waga

$$w(P(u, v)) = \sum_{e \in P(u, v)} w(e) \quad (8.2.2)$$

jest najmniejsza. Gdy wierzchołki u oraz v nie są połączone drogą (należą do różnych składowych grafu G), to przyjmujemy $P(u, v) = \infty$.

Dla sieci komunikacyjnych lub przepływowych wagi mogą reprezentować pewne wielkości fizyczne, takie jak koszt, odległość, efektywność, pojemność lub niezawodność, przypisane odpowiednim krawędziom. Prosty graf ważony może reprezentować jego macierz wag $W = [w_{ij}]$, gdzie w_{ij} jest wagą krawędzi e_{ij} między wierzchołkiem v_i oraz wierzchołkiem v_j . Wągom nieistniejących krawędzi nadaje się w zależności od zastosowań wartość ∞ lub 0.

Omówimy dalej dwa algorytmy służące do rozwiązywania problemu minimalnego drzewa spinającego. Są nimi algorytm Kruskala i algorytm Prima. Oba algorytmy są przykładami zachłannych algorytm-

mów optymalizacyjnych, które zostaną przedstawione bardziej ogólnie w rozdziale 11. W każdym kroku takiego algorytmu musimy dokonać jednego z wielu możliwych wyborów. Stosując strategię zachłanną, dokonujemy wyboru, który jest w danej chwili najlepszy.

Algorytm
Kruskala

*Algorytm Kruskala*³⁷ wyznaczający minimalne drzewo spinające ma następującą postać. Niech G będzie grafem o n wierzchołkach. Wtedy następujący algorytm daje rozwiązanie problemu minimalnego drzewa lub lasu spinającego:

Algorytm 8.2.1

1. Sortujemy krawędzie niemalejąco według wag.
2. Wybieramy krawędź e_1 o najmniejszej wadze.
3. Dodajemy krawędzie e_2, e_3, \dots, e_{n-1} , wybierając za każdym razem nową krawędź o najmniejszej możliwej wadze, która nie tworzy cyklu z dotychczas wybranymi krawędziami e_i .

Podgraf T grafu G , którego krawędziami są e_1, e_2, \dots, e_{n-1} , jest szukanym drzewem spinającym. Jeśli G jest grafem niespójnym, to algorytm Kruskala wyznacza minimalny las.

Algorytm
Prima

Metodą wyznaczania minimalnego drzewa spinającego grafu, która nie wymaga ani sortowania, ani sprawdzania istnienia cykli w każdym kroku, jest *algorytm Prima*⁵⁷, zwany też *algorytmem najbliższego sąsiada*. Dla grafu spójnego przebiega on w następujący sposób.

Algorytm 8.2.2

1. Niech $T = \emptyset$.
2. Wybieramy dowolny wierzchołek początkowy $v_1 \in T$ i dołączamy go do T .
3. Wybieramy dowolną krawędź incydentną z v_1 o najniższej wadze i drugi koniec tej krawędzi dołączamy do zbioru T .
4. Kolejno dla $i = 2, \dots, n - 1$ wybieramy dowolną krawędź e_i incydentną z dowolnym wierzchołkiem ze zbioru T o najniższej wadze taką, że jej drugi koniec nie należy do T , i dołączamy ten koniec do zbioru T .
5. Algorytm kończy się, gdy $T = V$.

Powyższa procedura wyznacza minimalne drzewo spinające zawierające krawędzie e_1, e_2, \dots, e_{n-1} . Najgorszy przypadek dla powyższego algorytmu pojawia się, gdy graf G jest grafem pełnym. W takiej

sytuacji w każdym kroku algorytmu musimy przeprowadzić maksymalną liczbę porównań, aby znaleźć najbliższy sąsiedni wierzchołek. Jeśli G jest grafem niespójnym, to dla wyznaczenia minimalnego lasu należy algorytm Prima zastosować dla każdej składowej oddzielnie.

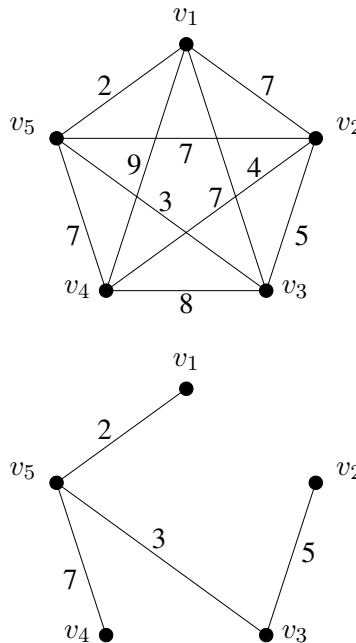
Algorytm zachłanny generujący minimalne drzewo spinające zawdzięczamy J. B. Kruskalowi (1956). Algorytm wyznaczający minimalne drzewo spinające metodą najbliższego sąsiada został odkryty w 1930 roku przez czeskiego matematyka V. Jarníka³², a następnie ponownie odkryty w 1957 roku przez R. C. Prima oraz niezależnie w 1959 roku przez holenderskiego informatyka E. Dijkstrę¹¹.

Przykład 8.2.1

Na rysunku 8.4 przedstawiony jest graf $G = K_5$ z wagami krawędzi. Oznaczmy $e_{ij} = (v_i, v_j)$. Krawędzie porządkujemy według rosnących wag:

$$e_{15}, e_{35}, e_{13}, e_{23}, e_{12}, e_{25}, e_{24}, e_{45}, e_{34}, e_{14}.$$

Zgodnie z algorytmem Kruskala wybieramy kolejne krawędzie, które nie tworzą cykli z poprzednio wybranymi. Są to krawędzie $e_{15}, e_{35}, e_{23}, e_{45}$. Waga tego drzewa wynosi $w(T) = 2 + 3 + 5 + 7 = 17$ i jest to minimalne drzewo spinające.



Rysunek 8.4. Graf do przykładu 8.2.1 i jego minimalne drzewo spinające

Stosując algorytm Prima, zaczynamy od dowolnego wierzchołka, na przykład od v_2 . Krawędź o najmniejszej wadze do pozostałych wierzchołków to e_{23} . Od zbioru wierzchołków $\{v_2, v_3\}$ do pozostałych prowadzi krawędź e_{35} , od zbioru $\{v_2, v_3, v_5\}$ do pozostałych prowadzi krawędź e_{15} , a w końcu wybieramy krawędź e_{45} . Dostaliśmy to samo drzewo co otrzymane za pomocą algorytmu Kruskala.

Drzewa minimalne mogą być różne w zależności od użytego algorytmu, sposobu posortowania przy algorytmie Kruskala (gdy wagi niektórych krawędzi są identyczne), wierzchołka startowego i kolejnych wyborów przy algorytmie Prima. Niemniej wagi tak otrzymanych drzew minimalnych są oczywiście takie same.

Długością ważoną drogi (u_0, u_1, \dots, u_n) w grafie $G(V, E)$ o nieujemnych wagach krawędzi (łuków) $e \in E$ jest suma wag krawędzi tej drogi. Do znajdowania najkrótszej drogi ważonej z ustalonego wierzchołka u w grafie $G(V, E)$ o nieujemnych wagach krawędzi $e \in E$ służy algorytm opracowany przez E. Dijkstrę.

*Algorytm
Dijkstry*

Algorytm ten znajduje w grafie wszystkie najkrótsze drogi pomiędzy wyróżnionym wierzchołkiem u a wszystkimi pozostałymi, dodatkowo wyliczając długość każdej z tych dróg. Algorytm Dijkstry jest również przykładem algorytmu zachłannego (zobacz na przykład [7], [13]). Sformułujemy go dla grafów nieskierowanych.

Algorytm 8.2.3

Tworzymy dwa zbiory wierzchołków Q i S . W trakcie algorytmu będziemy tworzyć funkcję $d(v) \geq 0$ określoną na V i funkcję $p(v)$ określoną na $V \setminus u$ o wartościach w V . Wierzchołek u jest jedynym wierzchołkiem początkowym.

1. Niech $Q = V, S = \emptyset$.
2. Dla każdego $v \in V \setminus u$ przyjmujemy $d(v) = \infty$ oraz $d(u) = 0$.
3. Funkcja $p(v)$ jest niezdefiniowana dla wszystkich $v \in V$.
4. Powtarzamy poniższe kroki algorytmu, aż $Q = \emptyset$.
 - (a) Wybieramy w Q wierzchołek o najmniejszym $d(v)$, usuwamy go z Q i dodajemy do S .
 - (b) Dla każdego $t \in \Gamma(v) \cap Q$, jeśli $d(t) > d(v) + w(v, t)$, to $d(t) = d(v) + w(v, t)$.
 - (c) Określamy $p(t) = v$.

W wyniku działania tego algorytmu ciąg wierzchołków $v, v_1 = p(v), v_2 = p(v_1), \dots, u = p(v_m)$ jest najkrótszą drogą z wierzchołka v do początkowego wierzchołka u .

Uwaga. Dla wybrania z Q wierzchołka v o najmniejszym $d(v)$ najlepiej przyjąć, że Q jest uporządkowany tak jak przy przeszukiwaniu grafu metodą BFS (algorytm 8.1.1).

Przykład 8.2.2

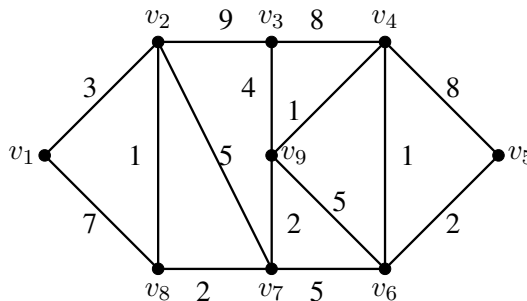
Dla grafu pokazanego na rysunku 8.5 najkrótszą drogą ważoną od v_1 do v_5 jest ciąg $(v_1, v_2, v_8, v_7, v_9, v_4, v_6, v_5)$. Ważona długość tej drogi jest równa 12, a długość nieważona jest równa 7. Długość zaś najkrótszej drogi nieważonej jest równa 4, a taką drogą jest na przykład ciąg v_1, v_2, v_3, v_4, v_5 . Długość ważona tej drogi wynosi 28.

Aby zastosować algorytm Dijkstry, ustalamy najpierw algorytmem BFS zbiór (kolejkę) Q (wiersz pierwszy) i nadajemy wartości z kroku 2 w algorytmie (wiersz drugi).

W kolejnych krokach algorytmu, idąc z kolejnych wierzchołków v z kolejki Q , wierzchołki $t \in V^+(v)$ grafu otrzymują wartości $d(t)$.

	v_1	v_2	v_8	v_3	v_7	v_4	v_9	v_6	v_5
	0	∞	∞	∞	∞	∞	∞	∞	∞
v_1	0	3	7	∞	∞	∞	∞	∞	∞
v_2	0	3	4	12	8	∞	∞	∞	∞
v_8	0	3	4	12	6	∞	∞	∞	∞
v_3	0	3	4	12	6	20	16	∞	∞
v_7	0	3	4	12	6	20	8	11	∞
v_9	0	3	4	12	6	9	8	11	∞
v_4	0	3	4	12	6	9	8	10	17
v_6	0	3	4	12	6	7	6	10	12
	v_1	v_3	v_2	v_8	v_9	v_7	v_4	v_6	

Wartości $p(v)$ dla $v \in V \setminus v_1$ są podane w ostatnim wierszu.



Rysunek 8.5. Graf do przykładu 8.2.2

11. Systemy i algorytmy zachłanne

11.1. Systemy zachłanne

Około 1980 roku J. Korte³⁵ i L. Lovász⁴² wprowadzili greedoidy, będące dalszym uogólnieniem matroidów, jako narzędzie do opisywania i badania algorytmów zachłannych*.

Niech E będzie zbiorem skończonym. Systemem zachłannym (greedoidem) nazywamy parę $M = (E, \mathcal{F})$ taką, że niepusta rodzina \mathcal{F} podzbiorów zbioru E spełnia następujące postulaty:

- (g_1) $\emptyset \in \mathcal{F}$,
- (g_2) jeśli $F_1, F_2 \in \mathcal{F}$, $|F_1| < |F_2|$, to istnieje $e \in F_2 \setminus F_1$ taki, że $F_1 \cup e \in \mathcal{F}$.

Zbiory
wykonalne
i osiągalne

Zbiory należące do \mathcal{F} nazywa się *wykonalnymi* (ang. *feasible*). Rodzinę \mathcal{F} nazywa się *osiągalną* (ang. *accessible*), gdy spełniony jest warunek (g_1) oraz

- (a_1) jeśli $F \in \mathcal{F} \setminus \emptyset$, to istnieje $e \in F$ takie, że $F \setminus e \in \mathcal{F}$.

Parę (E, \mathcal{F}) , gdzie \mathcal{F} jest rodziną osiągalną, nazywamy systemem osiągalnym. Greedoidy są systemami osiągalnymi. Łatwo też zauważyć, że każdy matroid jest systemem zachłannym, gdzie zbiorami wykonalnymi są zbiory niezależne matroidu.

Baza

Bazą systemu zachłannego nazywa się każdy maksymalny zbiór wykonalny. Z postulatu (g_2) wynika, że każda baza ma tę samą liczbę elementów.

Rząd

Rzędem $\rho(F)$ zbioru $F \subseteq E$ nazywa się liczbę elementów maksymalnego zbioru wykonalnego $F \subseteq A$. Zbiór $F \in \mathcal{F}$ jest wykonalny

*B. Korte, L Lovász, R. Schrader, *Greedoids*, Springer-Verlag, Berlin 1991. Patrz również B. Korte, J. Vygen, *Combinatorial Optimization, Theory and Algorithms*, Springer-Verlag, Berlin 2012.

wtedy i tylko wtedy, gdy $\rho(F) = |F|$. Każdy maksymalny wykonalny podzbiór F zbioru A ma tę samą liczbę elementów $\rho(A)$.

Przykład 11.1.1

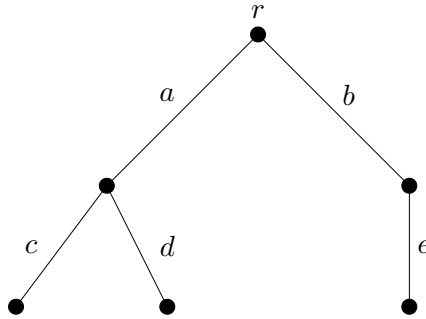
Niech $G = (V, E)$ będzie grafem nieskierowanym z wyróżnionym wierzchołkiem $r \in V$. Zbiory F krawędzi drzew (niekoniecznie spinających) zawierających wierzchołek r tworzą rodzinę \mathcal{F} taką, że (E, \mathcal{F}) jest greedoidem.

Przykład 11.1.2

Niech $T = (V, E, r)$ będzie drzewem o ustalonym korzeniu r . Określmy rodzinę \mathcal{F} jako rodzinę zbiorów krawędzi poddrzew drzewa T o tym samym korzeniu r . Jeśli $F \in \mathcal{F}$ oraz $F \neq \emptyset$, to usuwając krawędź e incydentną z wierzchołkiem stopnia 1, otrzymujemy znowu poddrzewo o korzeniu r lub zbiór pusty. Jediną bazą takiego greedoidu jest E (patrz zadanie 11.1).

Niech T będzie drzewem przedstawionym na rysunku 11.1. Wówczas $A \in \mathcal{F}$ tylko wtedy, gdy zajdzie jeden z przypadków:

- a) $\{a, b\} \subseteq A$,
- b) $a \in A, A \subseteq E \setminus \{b, e\}$,
- c) $b \in A, A \subseteq E \setminus \{a, c, d\}$,
- d) $a = \emptyset$.



Rysunek 11.1. Drzewo T z korzeniem r

Wtedy też:

$$\rho(A) = \begin{cases} |A|, & \text{jeśli } A \in \mathcal{F}, \\ 0, & \text{jeśli } A \notin \mathcal{F}. \end{cases}$$

W matroidzie każdy podzbiór bazy jest zbiorem niezależnym. Jeśli więc jedyną bazą jest cały zbiór E , to każdy zbiór $A \subseteq E$ jest niezależny, czyli jest to matroid wolny. Tutaj nie każdy podzbiór E należy do \mathcal{F} . Oznacza to (patrz zadanie 11.1), że tak skonstruowana para (E, \mathcal{F}) jest greedoidem, ale nie jest matroidem.

Wróćmy do warunków $(s_1) - (s_3)$ z twierdzenia 10.1.5 na str. 184 i zastąpmy warunek (s_4) warunkiem (s'_4) :

- (s_1) $A \subseteq \sigma(A)$,
- (s_2) jeśli $A \subseteq B$, to $\sigma(A) \subseteq \sigma(B)$,
- (s_3) $\sigma(\sigma(A)) = \sigma(A)$,
- (s'_4) jeśli $f \notin \sigma(A)$, $f \in \sigma(A \cup e)$, $f \neq e$ to $e \notin \sigma(A \cup f)$.

Antymatroid Parę (E, σ) spełniającą warunki $(s_1) - (s'_4)$ nazywamy antymatroidem. Antymatroid jest greedoidem, choć nie jest matroidem.

Przykład 11.1.3

Niech $\sigma(A)$ będzie minimalnym poddrzewem drzewa z przykładu 11.1.2, zawierającym krawędzie zbioru A . Wtedy $\sigma(A)$ spełnia warunki $(s_1) - (s'_4)$, czyli greedoid z tego przykładu jest antymatroidem.

Wprowadźmy trzy własności greedoidów o rodzinie wykonalnej \mathcal{F} .

- (v_0) Własność przedziałowa: jeśli $F, G, H \in \mathcal{F}$, $F \subseteq G \subseteq H$ oraz dla każdego $e \notin H$ zachodzi $F \cup e \in \mathcal{F}$, $H \cup e \in \mathcal{F}$, to wtedy $G \cup e \in \mathcal{F}$.
- (v_G) Własność przedziałowa bez ograniczenia górnego: jeśli $F, G \in \mathcal{F}$, $F \subseteq G$ oraz dla każdego $e \notin G$ zachodzi $F \cup e \in \mathcal{F}$, to wtedy $G \cup e \in \mathcal{F}$.
- (v_D) Własność przedziałowa bez ograniczenia dolnego: jeśli $G, H \in \mathcal{G}$, $F \subseteq H$ oraz $H \cup e \in \mathcal{F}$, to wtedy $G \cup e \in \mathcal{F}$.

Uzasadnieniem wprowadzenia tych pojęć są następujące trzy własności.

Własność 11.1.1

Greedoid mający własność przedziałową jest greedoidem, dla którego suma dwóch dowolnych zbiorów wykonalnych jest wykonalna, o ile zawiera inny zbiór wykonalny.

Własność 11.1.2

Greedoid mający własność przedziałową bez ograniczenia górnego jest antymatroidem.

Własność 11.1.3

Greedoid mający własność przedziałową bez ograniczenia dolnego jest matroidem.

Przykład 11.1.4

Niech $G = (V, E)$ będzie grafem prostym z wyróżnionym wierzchołkiem $r \in V$. Niech \mathcal{F} będzie rodziną zbiorów wierzchołków F takich, że $r \in F$ oraz F tworzy spójny podgraf grafu G . Wtedy (V, \mathcal{F}) jest antymatroidem. Ten greedoid jest zwany greedoidem przeszukiwań wierzchołków (ang. *vertex search greedoid*).

Zauważmy, że greedoid z przykładu 11.1.3 można uważać za szczególnie przypadek tak określonego greedoidu. Wystarczy do drzewa T z przykładu 11.1.3 dołączyć wierzchołek r_0 i dodać krawędź $\{r_0, r\}$, otrzymując drzewo T_0 , a następnie utworzyć graf krawędziowy $G = L(T_0)$ z wierzchołkiem $r = L(\{r_0, r\})$.

Przykład 11.1.5

Niech $G = (V, E)$ będzie digrafem prostym z wyróżnionym wierzchołkiem $r \in V$. Niech \mathcal{F} będzie rodziną łuków poddrzew skierowanych zawierających łuki skierowane zgodnie z drogami prowadzącymi od r do wierzchołków końcowych. (E, \mathcal{F}) jest greedoidem przedziałowym, ale nie jest ani matroidem, ani antymatroidem. Ten greedoid jest zwany greedoidem przeszukiwań łuków (ang. *line search greedoid*).

Zauważmy, że również tutaj greedoid z przykładu 11.1.3 można uważać za szczególnie przypadek tak określonego greedoidu. Wystarczy, że w drzewie T z przykładu 11.1.3 zorientujemy krawędzie tak, aby łuki były skierowane zgodnie z drogami prowadzącymi od r do wierzchołków końcowych.

Greedoid można też określić za pomocą poniżej zdefiniowanego języka zachłannego.

Niech E^* będzie zbiorem słów nad alfabetem E , to znaczy zbiorem ciągów o elementach (literach) z E . Konkatenacją $\alpha\beta$ słów $\alpha = x_1 \dots x_i$ oraz $\beta = y_1 \dots y_j$ jest słowo $x_1 \dots x_i y_1 \dots y_j$. Nośnikiem $\tilde{\alpha}$ słowa α jest zbiór różnych liter słowa α . Słowo jest *proste*, gdy każda litera występuje w słowie co najwyżej jeden raz.

Językiem nazywamy skończoną i niepustą rodzinę $\mathcal{L} \subseteq E^*$. Język jest prosty, gdy wszystkie jego słowa są proste. Nośnikiem $\tilde{\mathcal{L}}$ języka \mathcal{L} jest:

$$\tilde{\mathcal{L}} = \{\tilde{\alpha} : \alpha \in \mathcal{L}\}. \quad (11.1.1)$$

Język
zachłanny

Językiem zachłannym nad skończonym zbiorem E jest para (E, \mathcal{L}) , gdzie \mathcal{L} jest językiem prostym takim, że:

- (l₁) jeśli $\alpha = \beta\gamma$ i $\alpha \in \mathcal{L}$, to $\beta \in \mathcal{L}$,
- (l₂) jeśli $\alpha, \beta \in \mathcal{L}$ oraz $|\alpha| > |\beta|$, to α zawiera literę x taką, że $\beta x \in \mathcal{L}$.

Warunek (l_1) jest warunkiem (lewej) dziedziczności. Warunek (l_2) jest aksjomatem wymiany.

Twierdzenie 11.1.1

Systemy zachłanne i języki zachłanne są równoważne w następującym sensie.

- (i) Jeśli (E, \mathcal{L}) jest językiem zachłannym, to $(E, \tilde{\mathcal{L}})$ jest systemem zachłannym.
- (ii) Jeśli (E, \mathcal{F}) jest systemem zachłannym, to

$$\mathcal{L}(\mathcal{F}) = \{x_1 \dots x_k \in E^* : \{x_1, \dots, x_i\} \in \mathcal{F} \text{ dla } 1 \leq i \leq k\}$$
 jest językiem zachłannym.
- (iii) $\mathcal{L}(\tilde{\mathcal{L}}) = \mathcal{L}$ oraz $\tilde{\mathcal{L}}(\mathcal{F}) = \mathcal{F}$.

Przykład 11.1.6

Niech zbiorem liter będą krawędzie drzewa z przykładu 11.1.2. Określmy słowa języka \mathcal{L} jako ciągi symboli $\alpha = (x_1, \dots, x_k)$ takie, że zbiór $\{x_1, \dots, x_k\}$ jest drzewem o korzeniu r oraz jeśli $\alpha = \beta\gamma$, $\gamma = (y_1, \dots, y_l)$, to również $\{y_1, \dots, y_l\}$ jest drzewem o korzeniu r . Spełnione są więc warunki (l_1) oraz (l_2) .

11.2. Algorytmy zachłanne

Niech E będzie zbiorem skończonym oraz $w : E \rightarrow \mathbb{R}^+$. Wartość funkcji $w(e)$ nazywa się wagą elementu e . Niech \mathcal{S} będzie pewną rodziną podzbiorów zbioru E . Liczbę:

$$w(A) = \sum_{e \in A} w(e)$$

Algorytm
zachłanny

nazywa się wagą zbioru A . Rozważmy problem znalezienia zbioru $A \in \mathcal{S}$ o największej wadze. W tym celu sformułujemy następujący algorytm, zwany zachłannym.

Algorytm 11.2.1

1. Sortujemy elementy $e \in E$ według nierosnących wag w ciąg $e_1 \geq e_2 \geq \dots \geq e_n$ (lub według niemalejących wag $e_1 \leq e_2 \leq \dots \leq e_n$).
2. Zaczynając od $A = \emptyset$, dla $i = 1, \dots, n$, jeżeli $A \cup e_i \in \mathcal{S}$, to $A \leftarrow A \cup e_i$.

Otrzymany w ten sposób zbiór A jest wynikiem działania algorytmu zachłannego.

Algorytm zachłanny nie musi być optymalny, to znaczy zbiór $A \in \mathcal{S}$ znaleziony za pomocą algorytmu zachłannego nie musi mieć największej (najmniejszej) wagi.

Przykład 11.2.1

Niech:

$$A = \begin{bmatrix} 7 & 5 & 1 \\ 3 & 4 & 3 \\ 2 & 3 & 1 \end{bmatrix}.$$

Suma trzech elementów macierzy A wybranych algorytmem 11.2.1 takich, że z każdej kolumny może być tylko jeden element, jest największa i jest równa 15. Jeśli natomiast wybrane elementy nie mogą być z tych samych kolumn i wierszy, to suma takich trzech elementów nie jest największa. Algorytm zachłanny daje sumę 12, natomiast największa suma jest równa 13.

Bardziej ogólne przykłady pozostawimy jako zadania 11.2 i 11.3.

R. Rado⁵⁹ i J. Edmonds¹⁴ udowodnili poniższy wynik.

Twierdzenie 11.2.1

Jeżeli $M = (E, \mathcal{I})$ jest matroidem, to zbiór A wyznaczony przez algorytm zachłanny jest zbiorem niezależnym o największej (najmniejszej) wadze. Jeżeli (E, \mathcal{I}) nie jest matroidem, to istnieje waga $w : E \rightarrow \mathbb{R}^+$ taka, że A nie jest zbiorem o największej (najmniejszej) wadze.

Za pomocą algorytmu zachłannego i z wykorzystaniem twierdzenia 11.2.1 można wyznaczyć minimalne drzewo spinające grafu spójnego.

Jeżeli matroid jest grafowy, to algorytm Kruskala wyboru maksymalnego (minimalnego) drzewa spinającego w grafie G jest algorytmem zachłannym w matroidzie $M(G)$. Algorytm Prima nie jest zaś algorytmem zachłannym w matroidzie $M(G)$, gdyż w sformułowaniu tego algorytmu występują wierzchołki grafu. Jest on jednak algorytmem zachłannym w pewnym greedoidzie związanym z grafem G . Zagadnienie to omówimy w dalszej części tego punktu.

Przykład 11.2.2

W matroidzie Fano F_3 o elementach $\{1, 2, \dots, 7\}$ (zobacz rysunek 9.3 na str. 176) określimy wagę elementu i wzorem $w(i) = (i^2 \bmod 6) + 1$. Ponieważ różne elementy mogą mieć tę samą wagę, to elementy matroidu mogą być uporządkowane rosnąco na różne sposoby, na przykład:

1. (6, 5, 7, 1, 3, 2, 4),
2. (6, 5, 1, 7, 3, 2, 4),
3. (6, 7, 1, 5, 3, 4, 2).

Wybieramy bazę o najmniejszej wadze. Przy pierwszym i drugim uporządkowaniu minimalną bazę tworzą trzy pierwsze elementy, czyli $B_1 = \{5, 6, 7\}$ oraz $B_2 = \{1, 5, 6\}$. Przy uporządkowaniu trzecim element 1 będący na trzeciej pozycji należy do $\sigma(6, 7)$, więc musi być pominięty i wzięty element 5. Stąd $B_3 = B_1$.

Algorytm Huffmana

Algorytm Huffmana³¹ jest metodą bezstratnej kompresji danych, opracowaną w 1952 roku. Mimo że nie jest zbyt efektywny, stosuje się go ze względu na prostotę oraz brak ograniczeń patentowych. Jest przykładem algorytmu zachłannego.

Z symboli utworzony jest ciąg skończony (tekst) τ . Słowa kodowe o długości k bitów (znaków 0 i 1) każde wystarczają do zakodowania 2^k symboli. Na przykład powszechnie stosowany ośmiobitowy kod ASCII wystarcza na zakodowanie 128 symboli. Nie jest jednak konieczne, aby słowa kodowe były jednakowej długości.

Założmy, że każdemu symbolowi przypisujemy $s \in S$ i przyporządkujemy wagę $w(s)$. Niech $b(s)$ będzie liczbą bitów słowa kodującego symbol s . Kodowanie Huffmana polega na utworzeniu słów kodowych o niejednakowej długości w taki sposób, aby średnia ważona

$$W = \frac{\sum_{i=1}^n b(s_i) w(s_i)}{\sum_{i=1}^n w(s_i)} \quad (11.2.1)$$

była najmniejsza. Jeśli za wagę $w(s_i)$ przyjmiemy prawdopodobieństwo p_i występowania symbolu s_i w ciągu τ , to L określone wzorem (11.2.2) jest wartością oczekiwaną długości słowa kodowego:

$$L = \sum_{i=1}^n b(s_i) p(s_i), \quad (11.2.2)$$

gdyż mianownik prawej strony wzoru (11.2.1) jest równy 1 (patrz dodatek A.5).

Algorytm Huffmana:

1. Tworzymy listę drzew binarnych, które w wierzchołkach przechowują pary: (s_i, p_i) . Na początku drzewa składają się wyłącznie z korzenia.

2. Jeśli na liście jest więcej niż jedno drzewo, powtarzamy: usuwamy z listy dwa drzewa o najmniejszym prawdopodobieństwie zapisanym w korzeniu. Wstawiamy nowe drzewo, w którego korzeniu jest suma prawdopodobieństw usuniętych drzew, natomiast one same stają się jego lewym i prawym poddrzewem. Korzeń drzewa nie przechowuje symbolu.
3. Drzewo, które pozostanie na liście, jest nazywane drzewem Huffmana – prawdopodobieństwo zapisane w korzeniu jest równe 1, natomiast w liściach drzewa zapisane są symbole.

Algorytm Huffmana nie określa, w jakiej kolejności wybierać drzewa z listy, jeśli mają takie samo prawdopodobieństwo. Nie jest również określone, które z usuwanych drzew ma stać się lewym bądź prawym poddrzewem. Jednak bez względu na przyjęte rozwiązanie wartość oczekiwana długości kodu pozostaje taka sama.

Na podstawie drzewa Huffmana tworzone są słowa kodowe według następującego algorytmu:

1. Każdej lewej krawędzi drzewa przypisujemy 0, prawej 1 (można oczywiście odwrotnie).
2. Przechodzimy w głąb drzewa od korzenia do każdego liścia (symbolu): jeśli idziemy w prawo, dopisujemy do kodu bit o wartości 1; jeśli idziemy w lewo, dopisujemy do kodu bit o wartości 0.

Długość słowa kodowego jest równa głębokości symbolu w drzewie.

Przykład 11.2.3

Cztery symbole a, b, c, d występujące z prawdopodobieństwami:

a	b	c	d
0.1	0.2	0.3	0.4

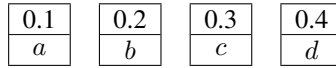
Proces budowy drzewa kodowego przedstawiony jest na rysunku 11.2. Kody znaków:

- $a = 000$,
- $b = 001$,
- $c = 01$,
- $d = 1$.

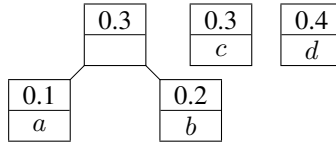
Wartość oczekiwana długości słowa kodowego wynosi:

$$3 \cdot 0.1 + 3 \cdot 0.2 + 2 \cdot 0.3 + 1 \cdot 0.4 = 1.9.$$

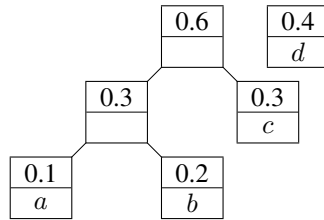
Cztery symbole – cztery drzewa składające się tylko z korzeni:



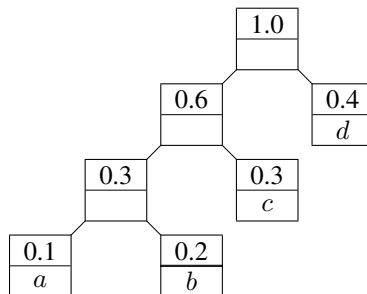
Łączymy korzenie z symbolami a oraz b :



Łączymy korzenie (a, b) oraz c :



Łączymy korzenie $((a, b), c)$ oraz d :



Rysunek 11.2. Budowa drzewa kodowego w algorytmie Huffmana

Więcej o algorytmie Huffmana i algorytmach na nim opartych można znaleźć w książkach [6] i [8].

Pozostaje pytanie, kiedy algorytm zachłanny jest optymalny. Z twierdzenia 11.2.1 wiadomo, że dla dowolnych funkcji wagi jest tak dla matroidów. Rząd matroidu jest funkcją submodularną. Mocniejszą własnością jest modularność funkcji. Funkcja $c(A)$, $A \subseteq E$, jest modularna, gdy spełnia równość:

$$c(A \cup B) = c(A) + c(B) - c(A \cap B).$$

Mocna własność wymiany

Mocną własność wymiany sformułujemy następująco. Dla każdego $A \in \mathcal{F}$, maksymalnego $B \in \mathcal{F}$, $A \subseteq B$, $x \in E \setminus B$, $A \cup x \in \mathcal{F}$ istnieje $x \in B \setminus A$ taki, że $A \cup y \in \mathcal{F}$ oraz $(B \setminus y) \cup x \in \mathcal{F}$.

Własność 11.2.1

Greedoid z przykładu 11.1.1 ma mocną własność wymiany.

O optymalności algorytmu zachłannego mówi poniższe twierdzenie.

Twierdzenie 11.2.2

Niech (E, \mathcal{F}) będzie greedoidem. Algorytm zachłanny znajduje zbiór $F \subseteq \mathcal{F}$ o maksymalnej wadze dla dowolnej modularnej funkcji $w : \mathcal{F} \rightarrow \mathbb{R}^+$ wtedy i tylko wtedy, gdy greedoid ma mocną własność wymiany.

Waga zbioru krawędzi grafu nieskierowanego określona jako suma wag jego krawędzi jest funkcją modularną. Z twierdzenia 11.2.2 i własności 11.2.1 wynika więc optymalność algorytmu Prima.

11.3. Zadania

11.1. Pokazać, że para (E, \mathcal{F}) skonstruowana w przykładzie 11.1.2 jest greedoidem.

11.2. Niech A będzie macierzą o n wierszach i m kolumnach o nieujemnych elementach. Uzasadnić, że suma m elementów macierzy wybranych za pomocą algorytmu 11.2.1 takich, że z i -tej kolumny, $1 \leq i \leq m$, może być tylko jeden element, jest największa.

11.3. Niech A będzie macierzą kwadratową $n \times n$ o nieujemnych elementach. Wybrane elementy muszą się znajdować w różnych wierszach i różnych kolumnach. Uzasadnić, że algorytm zachłanny nie musi być optymalny.

11.4. Uzasadnić stwierdzenie z przykładu 11.1.3.

11.5. Zbudować drzewo kodowe Huffmana, gdy symbole a, b, c, d występują z prawdopodobieństwami:

$$\begin{array}{c|c|c|c} a & b & c & d \\ \hline 0.1 & 0.1 & 0.1 & 0.7 \end{array},$$

i obliczyć wartość oczekiwaną L słowa kodowego.

11.6. Zbudować drzewo kodowe Huffmana dla pięciu symboli i prawdopodobieństw:

a	b	c	d	e
0.1	0.1	0.2	0.2	0.4

11.7. Zbudować drzewo kodowe Huffmana, gdy wszystkie z pięciu symboli mają to samo prawdopodobieństwo.

Skorowidz

A

aksjomat wymiany, 207
alfabet, 206
algorytm
 rozwiązywania równania
 rekurencyjnego, 75
Dijkstry, 155
Fleury'ego, 121
Huffmana, 209
Kruskala, 153, 208
najbliższego sąsiada, 153
najkrótszej drogi, 155
Prima, 153, 208, 212
przeszukiwania
 w głąb, 150
 wszerz, 149
 zachłanny, 154, 207
antyłańcuch, 14
antymatroid, 205
atom kraty, 172, 173, 176, 178

B

baza
 cykli, 134
 greedoidu, 203
 matroidu, 181, 186
 systemu zachłannego, 203
bijekcja, 27, 30
blok, 54, 57

C

centroid, 138

centrum, 107, 138
charakterystyka ciała
 skończonego, 170
ciało
 binarne, 170
 Galois, 170
 skończone, 169
 wielomianów, 170
ciąg
 arytmetyczny, 17
 geometryczny, 17
 rekurencyjny, 17
 zrównoważony, 80
cięcie
 grafu, 127, 131, 135
 minimalne, 131, 194
cięciwa grafu, 134
cykl
 Eulera, 120, 122, 144
 grafu, 86
 Hamiltona, 122
 matroidu, 182, 186
 permutacji, 33, 37
 prosty, 87

D

dendryt, 131
diagram
 Ferrersa, 68
 Hassego, 12
digraf, 88
 prosty, 88
 silnie spójny, 125

długość

 cyklu, 86
 permutacji, 33, 37
 drogi, 86
 słowa kodowego, 210

dopełnienie

 algebraiczne, 264
 grafu, 107, 144

dren, 157

droga, 86

 Eulera, 120, 144
 Hamiltona, 122
 minimalna, 152
 prosta, 86

drogi

 krawędziowo rozłączne, 128
 wierzchołkowo rozłączne, 128

drzewo, 87, 112, 115, 129

 binarne, 16, 136
 Huffmana, 210
 przeszukiwań, 149
 puste, 16, 136
 spinające, 131, 135, 186
 minimalne, 152

dziedzina funkcji, 27

E

element

 maksymalny, 12
 minimalny, 12
 najmniejszy, 12
 największy, 12
 elementy porównywalne, 12

F

fundamentalny zbiór
 cięć, 135
 cykli, 134
 funkcja, 27, 28, 57
 charakterystyczna, 263
 chromatyczna, 114
 modularna, 211
 na, 27, 66
 rozpięcia, 184
 różnowartościowa, 27, 29
 rzędu, 184
 submodularna, 184, 211
 tworząca, 72
 rząd, 196
 wykładnicza, 72
 wymierna, 265

G

gałąź drzewa, 134
 generator Fibonacciego, 23
 generowanie
 podzbiorów, 49
 podziałów zbioru, 55
 geometria
 afiniczna, 178, 185
 rzutowa, 175, 185, 197
 graf, 85
 acykliczny, 87, 129, 157
 bichromatyczny, 112
 dualny, 194
 dwudzielny, 105, 106, 112,
 114, 123, 142, 162
 regularny, 163
 eulerowski, 120, 125, 144
 hamiltonowski, 122, 144
 krawędziowy, 114
 kubiczny, 105
 k -kolorowalny, 111
 nieskierowany, 86, 152
 niezorientowany, 86
 orientowalny, 125
 pełny, 104, 112, 123, 132,
 137, 144
 dwudzielny, 106, 144
 Petersena, 105, 107, 112,
 118

planarny, 108, 109, 113,
 194
 płaski, 108
 półeulerowski, 120, 144
 półhamiltonowski, 144
 prosty, 86, 114, 137
 regularny, 105
 silnie spójny, 87
 skierowany, 88, 122, 125
 spójny, 87, 101, 106, 120,
 122, 127, 129, 137
 krawędziowo, 127
 ważony, 152
 zorientowany, 88, 122, 125
 greedoid, 203, 205
 przeszukiwań
 łuków, 206
 wierzchołków, 206
 grupa, 31
 abelowa, 32, 169, 199
 addytywna, 169
 multiplikatywna, 169
 permutacji, 33
 przemienna, 169
 skończona, 169
 gwiazda, 87

H

H -przepływ, 199

I

iloczyn
 kartezyjański, 262
 zbiorów, 262
 indeks chromatyczny, 113, 118
 indyktor, 263
 iniekcja, 27
 inwersja permutacji, 35, 42
 izomorfizm
 ciał skończonych, 170
 drzew, 136
 grafów, 102

J

język, 206
 prosty, 206
 zachłanny, 206, 207

K

klasa abstrakcji, 55, 263
 klika grafu, 107
 kobaza, 193
 kocykl, 131, 193
 kod
 Graya, 48, 122
 Huffmana, 209
 Prufera, 132, 145, 243
 kodrzewo, 131
 kombinacje, 44
 kongruencja, 262
 konkatenacja, 206
 kopęta, 193
 korzeń, 16, 87
 krata, 14, 178
 podprzestrzeni, 172
 rozdzielna, 14
 zupełna, 14
 krawędzie
 równoległe, 86
 sąsiednie, 89
 wielokrotne, 86
 krawędź, 86, 98, 106, 109, 113,
 136, 140
 kres
 dolny, 14
 górny, 14
 krok indukcyjny, 9, 11
 krotność elementu, 50
 kwantyfikatory
 ogólny, 261
 szczegółowy, 261

L

las, 129
 spinający, 131, 186
 minimalny, 152
 lemat o uściskach dłoni, 90
 liczba
 μ , 191
 chromatyczna, 111, 118
 cyklotomiczna, 131
 nieporządków, 62
 liczby
 Bella, 66, 82
 Catalana, 80, 136

Fibonacciego, 20, 25, 46,
75, 79, 100
uogólnione, 23
harmoniczne, 10, 17, 40
Lucasa, 25
Stirlinga
cykliczne, 39
drugiego rodzaju, 63,
71, 147
nieoznakowane, 39
pierwszego rodzaju, 38,
45, 52
podzbiorowe, 63
liczność zbioru, 51
liść, 87, 89, 138

Ł

łańcuch, 14
koniec, 14
nienasycony, 159
początek, 14
łuk, 88, 125
łuki wielokrotne, 88

M

macierz, 264
cykli, 94, 117, 135, 136
fundamentalna, 135
incydencji, 190
grafu, 91, 104, 116
grafu skierowanego, 93
sąsiedztwa, 97, 98, 116,
117, 125
stopni, 101
transponowana, 96
wag, 152
Matlab, 91
matroid, 181
baz, 181
bicykliczny, 188
binarny, 184, 186, 187
cykli, 182, 194
dualny, 192
Fano, 186, 193
grafowy, 186, 194
jednorodny, 184, 197
macierzowy, 185

ograniczenie, 186
reprezentowalny, 184
rozpięcie, 183
ściągnięcie, 186
transwersalny, 192
trywialny, 184
wolny, 184
z funkcją rzędu, 182
zbiorów niezależnych, 182
Maxima, 234
metryka, 107
minor
grafu, 108
matroidu, 187
moc zbioru skończonego, 27
most, 121, 127, 129
multigraf, 86
skierowany, 88
multizbiór, 50

N

następnik
lewy, 136
prawy, 136
nieporządek, 62
nierówność
Bernoulliego, 24
Weierstrassa, 24
niezależność liniowa, 263
niezawodność, 199
niezmiennik
izomorfizmu, 197
Tutte'a-Grothendiecka, 198
nośnik
języka, 206
słowa, 206

O

obraz zbioru, 27
obszar grafu, 109
obwód grafu, 87
Octave, 91
odległość, 263
odległość między
wierzchołkami, 107, 138
ograniczenie
matroidu, 186

ograniczenie zbioru
dolne, 14
górne, 14
orientacja grafu, 125

P

permanent macierzy, 35, 140
permutacja, 30
bez punktów stałych, 62
cykliczna, 33
element neutralny, 33
identycznościowa, 32
nieparzysta, 35
odwrotna, 33
parzysta, 35
transpozycja, 35
zapis znormalizowany, 38
pętla, 89, 96
grafu, 86, 88, 98
matroidu, 182
pętla wielokrotna, 86, 88
pierścień skończony, 169
płaszczyzna
afiniczna, 178
Fano, 176, 186
rzutowa, 176, 177
podgraf, 91
podłoga, 18
podmacierz, 264
podstawa indukcji, 9
podział
liczby, 67
sprzężony, 68
zbioru, 54
pokrycie
krawędziowe grafu, 141
wierzchołkowe grafu, 141
porządek
częściowy, 11, 24
leksykograficzny, 50
liniowy, 12
potęga
krocząca, 28
przyrastająca, 28, 30, 51,
52, 64, 78
ubywająca, 28, 29, 31, 38,
44, 64

problem kojarzenia
małżeństw, 143, 190

promień
drzewa, 139
grafu, 107
zbieżności, 265

przeciwdziedzina funkcji, 27

przeciwbraz zbioru, 27, 57

przekrój, 158
minimalny, 158
normalny, 158

przepływ, 157

przepustowość, 158

przestrzeń
liniowa, 171
metryczna, 107, 263

pseudolas, 188

Q

q -analog, 175

trójkąta Pascala, 180

R

reguła
sumowania po górnym
indeksie, 53
sumowania równoległego,
53

rekurencja, 15

relacja, 11, 27
antysymetryczna, 11
przechodnia, 11
przystawiana, 262
równoważności, 54, 178
symetryczna, 54
zwrotna, 11

rodzina
baz, 181
cykli, 182, 183
osiągalna, 203
zbiorów niezależnych, 182,
183

rozdrobienie podziału, 55

rozkład permutacji na cykle,
33, 39, 42

rozpięcie zbioru, 183

równanie charakterystyczne, 21

równanie rekurencyjne, 17, 75
liniowe, 19
jednorodne, 19
o stałych
współczynnikach, 19

rzędu k , 19

różnica
symetryczna, 88, 186, 262

zbiorów, 262

rząd
cięcia, 135
cykliczności, 131, 134
geometrii rzutowej, 176

grafu, 131

grupy, 169, 199

grupy skończonej, 199

macierzy, 191, 264
cykli, 136

matroidu, 182

permutacji, 34, 42

zbioru, 182, 203

S

Scilab, 91

sieć, 158
przepływowa, 157

silnia, 15, 31, 39, 43, 174
dolna, 28, 29, 31, 38, 44,
62, 64
górną, 28, 30, 51, 52, 64, 78

skierowanie grafu, 125

składowa
silnie spójna, 91
spójna, 91

skojarzenie, 140
doskonałe, 140
maksymalne, 140, 162
pełne, 140

słowo
kodowe, 209
proste, 206

splot, 73
Fibonacciego, 79

spójność
krawędziowa, 127
wierzchołkowa, 127

stopień wierzchołka, 89, 90,
112, 137, 138
wejściowy, 90, 94, 122
wyjściowy, 89, 94, 122

strumień, 157
maksymalny, 158

sufit, 18

suma
prosta matroidów, 187
zbiorów, 262

surjekcja, 27

symbol
Gaussa, 174
Newtona, 44, 174

system
osiągalny, 203
zachłanny, 203, 207

szereg
formalny, 72
odwrotny, 74
potęgowy, 265, 266

Ś

ściana grafu, 109

ściągnięcie
krawędzi, 108, 114
matroidu, 186

ściek, 157

średnia
teoretyczna, 266
ważona, 209, 266

średnica
drzewa, 139
grafu, 107

T

tekst, 209

tożsamość
Cassiniego, 21
Cauchy'ego, 47

transwersała, 189
częściowa, 189

trójkąt Pascala, 46

turniej, 125

twierdzenie
Brooksa, 112
Cauchy'ego, 38

Cayleya, 132
 Diraca, 124
 Eulera, 109
 Fermata, 53
 Forda-Fulkersona, 128, 159
 Gallaia, 141
 Halla, 143, 190
 Koniga, 114, 142
 Kuratowskiego, 108
 Landaua, 126
 Maclaurina, 265, 266
 Mengera, 128, 164
 Meyniela, 126
 o czterech barwach, 113
 o kojarzeniu małżeństw,
 142
 Orego, 124
 Redeiego, 125
 Rado, 192
 Rado-Edmondsa, 208
 Robbinsa, 126
 Thomassena, 126
 Vizinga, 112, 113
 typ permutacji, 33, 37, 38, 42

U

ujście, 157, 162
 ułamki proste, 265
 usunięcie
 krawędzi, 108, 114
 wierzchołka, 108

W

waga
 elementu, 207
 krawędzi, 152
 symbolu, 209
 wierzchołka, 138
 zbioru, 207
 warstwa, 178, 263
 warstwy funkcji, 57

wartość oczekiwana, 209, 266
 warunek
 brzegowy, 19
 początkowy, 11
 wejście, 157, 158
 wektor, 263
 wielomian
 chromatyczny, 114, 118,
 198, 200
 nierozkładalny, 170, 179
 przepływowy, 200
 Tutte'a, 196
 wierzchołek, 86, 98, 105, 109,
 111, 136, 140
 centralny, 138, 147
 centroidalny, 138, 147
 incydentny z krawędzią, 89
 izolowany, 89, 92
 końcowy, 89
 rozcinający, 127
 wiszący, 138
 wierzchołki
 incydentne, 88, 89
 sąsiednie, 89, 137
 własność
 przedziałowa, 205
 Steinitza, 181
 wymiany, 181
 mocna, 212
 współczynnik
 dwumianowy, 44, 77
 wielomianowy, 47
 wyjście, 158
 wymiar
 geometrii rzutowej, 176
 warstwy, 178
 wysokość drzewa, 87
 wyznacznik, 36, 264
 wzór
 Cayleya, 147, 244
 dwumianowy, 44

Newtona, 44
 Robbinsa, 31
 Stirlinga, 31

Z

zasada
 indukcji matematycznej,
 9, 11
 maksimum, 13
 minimum, 13
 szufladkowa Dirichleta, 55
 uogólniona, 58
 włączania-wyłączania, 59
 zbiór
 częściowo uporządkowany,
 11
 domknięty, 183
 krawędzi, 86, 114, 186
 łuków, 88
 niezależny, 182, 203
 permutacji, 31, 33
 rozdzielający
 graf spójny, 127
 wierzchołki, 128
 rozspajający
 graf spójny, 127
 wierzchołki, 128
 wewnętrznie stabilny, 140
 wierzchołków, 86, 114
 wykonalny, 203
 z powtórzeniami, 50
 złoty podział, 22
 złożenie
 permutacji, 32, 35
 transpozycji, 36
 zmienna losowa, 266
 znak permutacji, 35, 37

Ź

źródło, 157, 162

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

Na zadane w ankiecie pytanie o definicję matematyki dyskretnej pewien student Politechniki Gdańskiej odpowiedział, że jest to dział matematyki, który „dyskretnie wciska się, gdzie się da”. Choć prawdopodobnie nie o taką odpowiedź chodziło pytającemu, z pewnością jest w niej trochę prawdy. Z matematyką dyskretną mamy obecnie do czynienia dosłownie wszędzie, ponieważ wszędzie obecna jest informatyka, która wykorzystuje wiele pojęć i konstrukcji powstałych właśnie dzięki temu stosunkowo mało znanemu działowi królowej nauk.

Matematyka dyskretna to zbiorcza nazwa różnych działów matematyki zajmujących się badaniem struktur nieciągłych, a więc takich, które w naturalny sposób znajdują zastosowanie w informatyce. Kryptografia, teoria gier i teoria grafów — to tylko niektóre z działów matematyki dyskretnej praktycznie wykorzystywane przez wielu programistów w codziennej pracy. A jeśli doda się do nich takie zagadnienia jak rekurencja czy algorytmy zachłanne, potrzeba zrozumienia podstaw tego działu matematyki staje się chyba jasna dla wszystkich adeptów informatyki.

Ten podręcznik powstał na bazie doświadczeń autorów w prowadzeniu zajęć z matematyki dyskretnej, teorii grafów i algorytmów kombinatorycznych na Politechnice Wrocławskiej, na Wydziale Podstawowych Problemów Techniki, na Uniwersytecie Ekonomicznym w Poznaniu, na Wydziale Informatyki i Gospodarki Elektronicznej, oraz w Państwowej Wyższej Szkole Zawodowej im. Witelona w Legnicy, na Wydziale Nauk Technicznych i Ekonomicznych. Zajęcia były prowadzone dla studentów informatyki, a także dla tych z kierunku informatyka i ekonometria. I to przede wszystkim dla studentów kierunków informatycznych przeznaczona jest ta książka. Zawiera ona również wiadomości bardziej zaawansowane, przydatne dla doktorantów i zaawansowanych programistów, którym daje teoretyczne podstawy do studiowania algorytmów.

- Indukcja i rekurencja
- Rozmieszczenia i permutacje
- Kombinacje i podziały
- Grafy i drzewa
- Algorytmy grafowe
- Struktury kombinatoryczne
- Systemy i algorytmy zachłanne

Matematyka dyskretna bez tajemnic



Helion



helion.pl



0 801 339900



0 601 339900

Sprawdź nasze szkolenia!



AKADEMIA IT & BUSINESS

WWW.SZKOLENIA.HELION.PL

KOD KORZYŚCI
Sięgnij po więcej! ▶



ISBN 978-83-283-4011-4



9 788328 340114

INFORMATYKA W NAJLEPSZYM WYDANIU

Cena: 49,00 zł