AUTOMATYKA • 2011 • Tom 15 • Zeszyt 2

Łukasz Rojek*, Konrad Wala*

Metaheurystyka pszczela w kolorowaniu wierzchołków grafu

1. Wprowadzenie

Ze względu na szerokie spektrum zastosowań, problemy silnie NP-trudne pozostają interesującym obszarem badawczym zarówno dla specjalistów z matematyki dyskretnej, jak i informatyków. Bardzo często badane są tu, pod kątem efektywności procesu poszukiwania, metaheurystyki inspirowane przez naturę. Obserwacja biologicznych układów emergentnych, takich jak roje owadów, pozwoliła na odkrycie statystycznych procesów samoorganizacji wykorzystanych m. in. do optymalizacji dyskretnej. Począwszy od roku 2005, praca [1] wykonana w Manufacturing Engineering Centre, Cardiff University, UK, badana jest metaheurystyka pszczela należąca do podejść rojowych (por. [2–5]). Powstaje tu pytanie, czy możemy spodziewać się w przypadku stosowania algorytmów będących konkretnymi realizacjami tej metaheurystyki lepszych rezultatów optymalizacji silnie trudnych problemów kombinatorycznych w porównaniu do takich, klasycznych już w tej chwili i numerycznie przebadanych metaheurystyk, jak genetyczna czy ewolucyjna oraz mrówkowa. Wśród trudnych problemów kombinatorycznej optymalizacji szczególne miejsce zajmują problemy permutacyjne, gdzie rozwiązaniem problemu jest kolejność wykonywania określonych zadań czy czynności, a także kolejność przydziału specyficznych (z zakresy wąskiego gardła) zasobów.

Kolorowanie grafów, w ogólnym przypadku silnie NP-trudny problem optymalizacji kombinatorycznej, jest ciągle interesującym problemem badawczym dla wielu specjalistów [6–8]. Klasyczny, wierzchołkowy problem kolorowania grafu polega na przypisaniu wierzchołkom grafu jednego z kolorów (nazwy, liczby porządkowej) tak, ażeby żaden z sąsiednich wierzchołków nie miał tego samego koloru. Przyjęto, że wierzchołkowym dopuszczalnym k-kolorowaniem grafu $G = (V, E)$ jest surekcja x: $V \to \{1, 2, ..., k\}$, gdzie $x(i) \neq x(j)$ jeżeli $\{i, j\} \in E$. Problem optymalizacji polega na wyznaczeniu pokolorowania x minimalizującego liczbę zastosowanych kolorów: $k \to \min$.

* Wyższa Szkoła Biznesu w Dąbrowie Górniczej

Historycznie, profesor uniwersytetu w Cambridge A. Cayley w roku 1879 postawił pierwszy problem kolorowania wierzchołków grafu na spotkaniu Londyńskiego Towarzystwa Matematycznego, w celu ustalenia sposobu kolorowania map administracyjnych minimalną liczbą kolorów. W tym samym roku jego uczeń A. Kempe podał dowód twierdzenia, iż 4 kolory wystarczą do pokolorowania dowolnej dwuwymiarowej mapy. Jednak w 1890 matematyk P. Heawood wskazał błędy w dowodzie Kempe'a, podając własny dowód dotyczący kolorowania za pomocą 5 barw. Poprawny dowód dla czterech kolorów przeprowadzili dopiero w 1976 roku K. Appel i W. Haken polegający na rozpatrzeniu 1936 typów map, wspomagany obliczeniami komputerowymi.

Sytuacje w których należy unikać określonych konfliktów pomiędzy obiektami, są często modelowane jako problemy kolorowania grafów. Jeśli do wykonania zadań i oraz j reprezentowanych przez wierzchołki grafu w konkretnym przedziale czasu potrzebne są zbiory środków $S(i)$ i $S(j)$ oraz ma miejsce konflikt $S(i) \cap S(j) \neq \varnothing$, to łącząc wierzchołki i, j krawędzią $\{i, j\}$ zapewniamy tym zadaniom, w dopuszczalnym pokolorowaniu, inne przedziały czasu realizacji, a w optymalnym pokolorowaniu minimalną liczbę przedziałów realizacji wszystkich planowanych zadań.

2. Metaheurystyka i algorytm pszczeli

2.1. Mataheurystyka pszczela

Dla wygody Czytelnika, poniżej zaprezentowano krótki opis metaheurystyki pszczelej inspirowanej przez organizację pszczół miodnych poszukujących pożywienia; streszczenie opracowano na podstawie [1]. Podstawową cechą organizacji żerowania pszczół naśladowaną w metaheurystyce jest stosowanie i przetwarzanie populacji rozwiązań w każdej iteracji, przy czym wyjściem każdej iteracji jest także populacja rozwiązań. Parametry metaheurystyki:

m – liczba wybranych lokalizacji zagonów kwiatowych z *n* odwiedzonych przez zwiadowców,
e – liczba najlepszych, elitarnych, lokalizacji zagonów wybranych z *m*,
nep – liczba pszczół rekrutowanych dla *e* najlepszych lokalizacji,
nsp – liczba pszczół rekrutowana dla pozostałych (*m*-*e*) wybranych lokalizacji,

Pseudokod metaheurystyki pszczelej.

1. Inicjalizuj populację *n* rozwiązań losowych i oceń ich jakość (fitness).
2. while (nie wystąpi kryterium stopu) //Formowanie nowej populacji.
3. Wybierz *m* lokalizacji do przeszukiwania sąsiedztwa.
4. Rekrutuj pszczoły do wybranych lokalizacji (więcej pszczół – *nep* – dla najlepszych *e* lokalizacji i mniej – *nsp* – dla pozostałych (*m*-*e*)) i oceń ich jakość.
5. Wybierz najlepszą pszczołę z każdego zagonu kwiatowego
6. Przydziel pozostałe pszczoły do losowego szukania i oceń ich jakość.
7. end while

Poszukiwania rozpoczynają się od losowego umieszczenia *n* zwiadowców w zbiorze rozwiązań i oceny ich jakości. W kroku 3, pszczoły o najwyższej jakości są wytypowane jako „pszczoły wybrane" i lokalizacje odwiedzone przez nie są wybrane do **pełnego** przeszukiwania sąsiedztwa. Następnie, w krokach 4 i 5, algorytm prowadzi poszukiwania w sąsiedztwie wybranych lokalizacji, przydzielając więcej pszczół do przeszukiwania w pobliżu *e* najlepszych lokalizacji. Pszczoły mogą być wybrane bezpośrednio zgodnie z ich jakością związaną z lokalizacjami, które odwiedzają. Alternatywnie, wartości jakości są zastosowane do określenia prawdopodobieństwa wyboru pszczół. Poszukiwanie w sąsiedztwie *e* najlepszych lokalizacji, które reprezentują bardziej obiecujące rozwiązania, jest wykonywane bardziej szczegółowo przez rekrutację większej liczby pszczół, *nep*, aniżeli dla pozostałych wybranych (*m*-*e*) – *nsp*, *nsp* < *nep*.Ta różnicowana rekrutacja jest kluczową operacją algorytmu pszczelego.

W kroku 5, z każdego zagonu kwiatowego tylko pszczoła z najwyższą oceną jakości jest wybierana do następnej populacji. W naturze nie ma takiego ograniczenia. Zostało ono wprowadzone w celu redukcji punktów eksploracji. W kroku 6 pozostałe pszczoły populacji są przydzielone losowo w przestrzeni rozwiązań, przeszukując nowe potencjalne rozwiązania.

2.2. Algorytm pszczeli

Poniżej przedstawiono badany algorytm pszczeli AP uściślający funkcje metaheurystyki pszczelej dla problemów optymalizacji kombinatorycznej, gdzie zagony kwiatowe są definiowane jako sąsiedztwa poprawianych rozwiązań. Parametry algorytmu:

pocz – początkowa liczba pszczół zwiadowców,
m – liczba wybranych lokalizacji z *n* odwiedzonych przez zwiadowców,
e – liczba najlepszych (elitarnych) lokalizacji wybranych z *m*,
nep – liczba pszczół rekrutowanych dla *e* najlepszych lokalizacji,
nsp – liczba pszczół rekrutowana dla pozostałych (*m*-*e*) wybranych lokalizacji,
k – liczba określająca wielkość przeszukiwanego sąsiedztwa.

Pseudokod algorytmu pszczelego AP
dla problemów optymalizacji kombinatorycznej

1. Inicjalizuj populację *P* składającą się z *pocz* losowo wybranych rozwiązań z przestrzeni szukania, oceń ich jakość (fitness) i zapamiętaj najlepsze jako *best* oraz ustal wartość k ($k \geq 1$).
2. Z populacji *P* wybierz *m* najlepszych rozwiązań, zbiór $M = \{1, ..., m\}$, ze zbioru *M* wybierz *e* najlepszych – elitarnych – rozwiązań, zbiór $EL = \{1, ..., e\}$, i zapamiętaj najlepsze rozwiązanie zbioru *EL* jako *bestEL* , gdzie $EL \subset M$ oraz $e < m\text{-}e < m < pocz$. Wyznacz najlepsze aktualne rozwiązanie $best := \arg\min \{Q(best), Q(bestEL)\}$.
3. Jeżeli jest spełniony warunek zatrzymania obliczeń (np. wykonanie zadanej liczby iteracji, wykonanie zadanej liczby iteracji bez poprawy rozwiązania *best*) to STOP; wyjście: *best*, *Q*(*best*).

4. Dla każdego $i \in EL$ z sąsiedztwa $W_k(i)$ wybierz losowo e_i rozwiązań, gdzie $\Sigma_i \in {}_{EL}\, e_i = nep$. Dla każdego $i \in \{M\text{-}EL\}$ z sąsiedztwa $W_k(i)$ wybierz losowo m_i rozwiązań, gdzie $\Sigma_i \in {}_{\{M\text{-}EL\}}\, m_i = nsp$ oraz $nsp < nep$; wybrane rozwiązania tworzą nową populację P. Przejdź do kroku 2. // Formowanie nowej populacji: rekrutacja pszczół do wybranych lokalizacji (więcej pszczół dla najlepszych *e* lokalizacji).

Algorytm rozpoczyna od losowego umieszczenia *pocz* zwiadowców w przestrzeni poszukiwań i ocenie jakości odwiedzonych lokalizacji. W kroku 2 pszczoły o najwyższej jakości są wytypowane jako „pszczoły wybrane" i lokalizacje odwiedzone przez nie są wybrane do przeszukiwania sąsiedztwa. Następnie, w kroku 4 algorytm prowadzi poszukiwania w sąsiedztwie wybranych lokalizacji, przydzielając więcej pszczół do przeszukiwania w pobliżu *e* najlepszych lokalizacji. Poszukiwanie w sąsiedztwie *e* najlepszych lokalizacji, które reprezentują bardziej obiecujące rozwiązania, jest wykonywana bardziej szczegółowo przez rekrutację większej liczby pszczół aniżeli dla pozostałych wybranych (*m*-*e*); **ta różnicowana rekrutacja jest kluczową operacją algorytmu pszczelego**.

Definicje sąsiedztwa $W_k(i)$ dla $i \in P$ w przypadku optymalizacji kombinatorycznej, gdzie rozwiązaniem jest *n*-elementowa permutacja π. Niech $\pi = (\pi(1), ..., \pi((i), ..., \pi(j), ..., \pi(n))$ jest permutacją zbioru *n* elementowego. Przyjęto, że rozwiązanie π^1 sąsiednie do π jest wyznaczone przez operację $ruch(\pi, i, j)$, $i, j \in \{1, ..., n)$ polegającą na przestawieniu elementów $\pi(i)$ i $\pi(j)$ permutacji π, tj.: $\pi^1 = (\pi(1), ..., \pi(j), ..., \pi(i), ..., \pi(n)) = ruch(\pi, i, j)$.

Sasiedztwo $W_k(\pi)$, $k = 1,2, ...$ jest to zbiór wszystkich rozwiązań otrzymanych z rozwiązania π przez *k* operacji *ruch*:

$$W_k(\pi) = \{\pi^k : \pi^k = ruch(\pi^{k-1}, i, j),\ \ i, j \in \{1, ..., n\}\}$$

$$\pi^{k-1} = ruch(\pi^{k-2}, i, j),\ \ i, j \in \{1, ..., n\}\}, k = 1, 2, ...$$

Wybór losowy rozwiązań z sąsiedztwa W_k oznacza, że w każdym z *k* ruchów liczby *i*, *j* są losowo wybierane; rozkład równomierny, ze zbioru $\{1, ..., n\}$.

W badanym algorytmie AP wprowadzono pewne modyfikacje w stosunku do klasycznej już metaheurystyki, gdzie przeszukiwane jest pełne sąsiedztwo rozwiązań zbiorów *E* i M-*E*. W propozycji zastosowano sąsiedztwo z parametrem *k*, $W_k()$, którego wartość, a więc i liczność sąsiedztwa, jest parametrem algorytmu i podczas wstępnych badań numerycznych może być dobierane przez użytkownika. Z każdego sąsiedztwa wybierana jest losowo ustalona liczba rozwiązań, można więc przeszukiwać różne sąsiedztwa w rozsądnym czasie, sprawdzając tylko niedużą liczbę rozwiązań w każdym. Dodatkowo, przeszukiwanie losowe większych sąsiedztw umożliwia likwidację kroku 6 metaheyrystyki, a przez to uproszczenie algorytmu, szczególnie w zakresie doboru jego parametrów eksploatacyjnych. Zauważmy jeszcze, ze realizacja kroku 6 jest niezbedna dla roju pszczół do adaptacji procesu poszukiwania nowych zagonów kwiatowych w związku ze zmieniającą się porą kwitnienia różnych gatunków. Wydaje się, że w przypadku poszukiwania dobrych rozwiązań problemów kombinatorycznych, problemów statycznych, wystarczy na początku obliczeń wygenerować dostatecznie dużą populację początkową.

3. Kolorowanie wierzchołków grafu

Problem kolorowania wierzchołków grafu prostego $G = (V, E)$ jest w ogólnym przypadku silnie NP-trudny [6, 7], wobec tego jest merytorycznie uzasadnione stosowanie wielomianowych algorytmów heurystycznych, wśród których ważne miejsce zajmują algorytmy permutacyjne, gdzie wierzchołki zbioru V są uszeregowane w permutację $\pi = (\pi(1), ..., \pi(j), ..., \pi(n))$, $\pi(j) \in V$. Permutacja π określa rozwiązanie problemu kolorowania, k-kolorowanie $k = k(\pi)$, w sposób jednoznaczny, jeżeli przyjąć regułę kolorowania polegającą na przydziale kolejnym wierzchołkom permutacji najmniejszego dopuszczalnego koloru. Dokładnie: $x(\pi(1)) = 1$, dla $j = 2, 3, ..., n$: $x(\pi(j)) = \min \{k : k \geq 1, x(\pi(i) \neq k$ dla $\{\pi(i), \pi(j)\} \in E$, $i = 1, 2, ..., j–1\}$.

Dla dowolnego grafu G można znaleźć taką permutację π, ze wykonanie wymienionej powyżej reguły kolorowania prowadzi do pokolorowania optymalnego. Wystarczy w tym celu wziąć jedno z pokolorowań optymalnych x i szeregować wierzchołki grafu w kolejności niemalejących kolorów $x(\pi(i)) \leq x(\pi(j))$ dla $i < j$. Dla takiego uszeregowania pokolorowanie y wyznaczone przez regułę kolorowania spełnia nierówność $y(\pi(i)) \leq x(\pi(i))$, dla $1 \leq i \leq n$, a więc jest to także pokolorowanie optymalne. Celem algorytmów permutacyjnych kolorowania jest wyznaczenie permutacji numerów wierzchołków grafu która, po zastosowaniu reguły kolorowania, wyznacza optymalne lub bliskie do optymalnego pokolorowanie wierzchołków grafu.

Podstawowe algorytmy konstrukcyjne szeregowania wierzchołków zbioru V w permutacje π to:

- Algorytm LF (*Largest-First*), gdzie wierzchołki są szeregowane w kolejności nierosnących stopni $deg(j)$ wierzchołków $j \in V$, tj.: $deg(\pi(j+1)) \geq deg(\pi(j))$, $j = 1, 2, ..., n–1$.
- Algorytm SL (*Smallest-Last*) zaproponowany przez D. Matule.
- Algorytm SLF (*Saturated Largest First*), w którym kolorowanie wierzchołków odbywa się na podstawie stopnia nasycenia wierzchołka, czyli liczby różnych kolorów, użytych do pokolorowania jego sąsiadów.

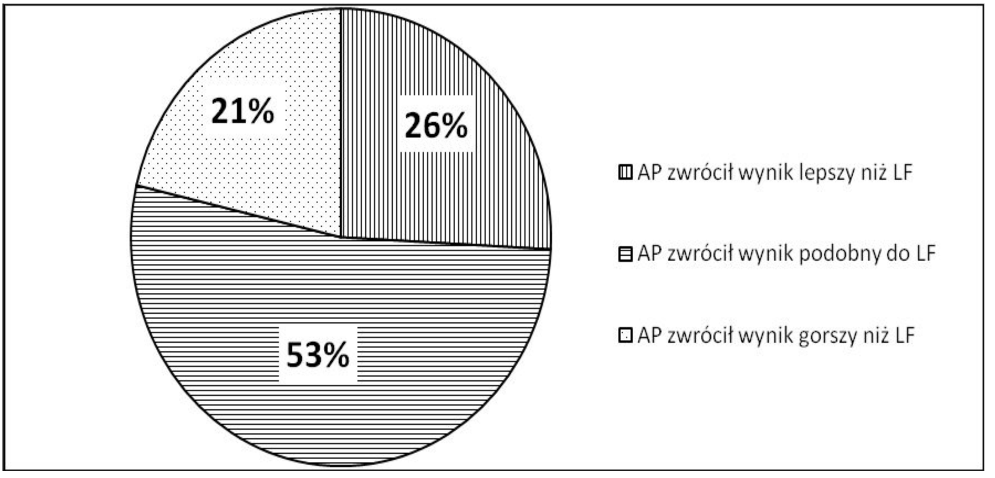
Zauważmy, że powyższe algorytmy konstruują uszeregowanie, w czasie wielomianowym, na podstawie lokalnej informacji o strukturze grafu, jaką jest stopień wierzchołków. Dla każdego z nich można wskazać przykłady, dla których uzyskanie pokolorowania optymalnego jest niemożliwe. Losowy algorytm permutacyjny, gdzie szeregowanie wierzchołków wykonywane jest losowo, ma szansę wyznaczyć dla dowolnego grafu pokolorowanie optymalne, ale średnio liczba zastosowanych kolorów przez ten algorytm jest większa niż przez algorytmy LF, SL, SLF [6, 7]. W następnym punkcie przedstawiono wyniki badań komputerowych zastosowania algorytmu pszczelego do kolorowania, porównując je z najprostszym algorytmem konstrukcyjnym LF.

4. Eksperymenty komputerowe

W tabeli 1 przedstawiono wybrane wyniki badań komputerowych omówionego algorytmu AP wykonane dla losowo wygenerowanych grafów prostych z liczbą wierzchołków n = 100 i dodatkowo scharakteryzowanych przez gęstość krawędzi GE. W wykonanych eksperymentach obliczeniowych populacja początkowa składała się z 500 wygenerowanych losowo permutacji. Pozostałe parametry algorytmu AP były następujące: m = 30, e = 14, nep = 70, nsp = 30 oraz k = 5, warunkiem stopu algorytmu było wykonanie 3. iteracje bez poprawy liczby kolorów. Na rysunku 1 przedstawiono procentowe porównanie wyników zamieszczonych w tabeli 1.

Tabela 1
Zestawienie liczby kolorów użytych przez algorytmy pszczeli AP
oraz konstrukcyjny LF, gdzie GE – gęstość krawędzi grafu

Lp.	GE	AP	LF
1	**0,57**	24	23
2	**0,57**	25	23
3	**0,55**	25	24
4	**0,53**	24	22
5	**0,53**	23	21
6	**0,52**	23	24
7	**0,52**	23	22
8	**0,51**	22	23
9	**0,50**	23	23
10	**0,50**	22	20
11	**0,50**	22	22
12	**0,49**	21	21
13	**0,48**	21	20
14	**0,47**	21	21
15	**0,46**	10	10
16	**0,45**	20	19
17	**0,45**	18	19
18	**0,42**	19	18
19	**0,40**	18	18
20	**0,25**	11	12
21	**0,25**	11	11
22	**0,24**	11	11
23	**0,24**	12	12
24	**0,24**	11	12
25	**0,23**	11	11
26	**0,23**	11	11
27	**0,23**	10	11
28	**0,23**	11	11
29	**0,23**	10	10
30	**0,23**	11	10
31	**0,22**	11	11
32	**0,22**	11	11
33	**0,22**	10	11
34	**0,22**	11	11
35	**0,22**	11	10
36	**0,22**	10	11
37	**0,21**	10	11
38	**0,21**	10	10
39	**0,21**	10	11
40	**0,21**	10	11
41	**0,21**	10	10
42	**0,21**	10	11
43	**0,21**	10	10
44	**0,21**	10	10
45	**0,21**	10	10
46	**0,21**	11	11
47	**0,21**	10	10
48	**0,20**	10	10
49	**0,20**	10	9
50	**0,20**	10	11
51	**0,20**	10	10
52	**0,20**	10	10
53	**0,20**	10	10
54	**0,20**	10	9
55	**0,20**	10	10
56	**0,20**	9	10
57	**0,20**	9	10
58	**0,20**	10	9
59	**0,20**	10	10
60	**0,20**	10	9
61	**0,19**	9	10
62	**0,19**	10	10
63	**0,19**	9	9
64	**0,19**	10	10
65	**0,19**	9	10
66	**0,19**	10	9
67	**0,19**	9	10
68	**0,19**	10	9
69	**0,19**	9	10
70	**0,19**	9	9
71	**0,19**	9	9
72	**0,19**	9	9
73	**0,18**	10	9
74	**0,18**	9	10
75	**0,18**	9	9
76	**0,18**	9	9
77	**0,18**	9	9
78	**0,17**	9	10
79	**0,17**	9	9
80	**0,17**	9	9
81	**0,17**	9	9
82	**0,17**	9	9
83	**0,17**	9	9
84	**0,17**	8	9
85	**0,17**	9	9
86	**0,17**	9	9
87	**0,17**	8	9
88	**0,16**	9	9
89	**0,16**	9	9



Rys. 1. Procentowe porównanie wyników badań algorytmu AP z algorytmem LF

5. Zakończenie

W pracy zaproponowano algorytm pszczeli AP dla permutacyjnych problemów optymalizacji kombinatorycznej i przeprowadzono wstępne badania numeryczne jego efektywności w kolorowaniu wierzchołków grafu prostego. Wyniki uzyskane w kolorowaniu losowo wygenerowanych grafów porównano z wynikami standardowego algorytmu konstrukcyjnego LF.

Odnotujmy, że algorytmy pszczele bazują tylko na przeszukiwaniu sąsiedztwa, charakteryzując się dodatkowo regułą: intensywniej przeszukuj sąsiedztwa lepszych rozwiązań. Algorytmami z przeszukiwaniem tylko sąsiedztwa poprawianych rozwiązań są między innymi, od wielu już lat eksploatowane, algorytmy MSLS (*multiple start local search*) i VNS (*variable neighbourhood search*) i trudno spodziewać się uzyskania od nich lepszych wyników szukania. Wydaje się, że do silnie trudnych problemów optymalizacji kombinatorycznej lepiej jest stosować algorytmy hybrydowe typu GLS (*genetic local search*) czy GTS (*genetic tabu search*), gdzie przeszukiwanie sąsiedztwa jest wspierane przez operatory krzyżowania, których wynalezienie zdecydowanie przyśpieszyło w przyrodzie proces ewolucji.

Literatura

[1] Pham D.T., Ghanbarzadeh A., Koc E., Otri S., Rahim S., Zaidi M., *The Bee Algorithm – A novel tool for complex optimization*. Technical Note, Manufacturing Engineering Centre, Cardiff University, UK, 2005; Published by Elsevier Ltd. 2006.

[2] Filipowicz B., Chmieiel W., Kadłuczka P., *Ukierunkowane przeszukiwanie przestrzeni rozwiązań w algorytmach rojowych*. Automatyka (półrocznik AGH), t. 13, z. 2, 2009, 247–255.

[3] Karaboga D., Akay B., *A comparative study of Artifical Bee Colony algoritm*. Applied Mathematics and Computation, 214, 2009, 108–132.
[4] Rojek Ł., *System informatycznydo do badania algorytmu pszczelego na przykładzie problemu kolorowania wierzchołków grafu prostego*. Dyplomowa praca magisterska wykonana w Wyższej Szkole Biznesu, Dąbrowa Górnicza 2011, 63, promotor: K.Wala.
[5] [on-line], 14.03.2001 http://en.wikipedia.org/wiki/Bees_algorithm.
[6] Agnarsson G., Greenlaw R., *Graph theory: Modeling, applications, and algorithms*. Pearson Education 2007, Inc.
[7] Diestel R., *Graph theory*. Springer-Verlag, NY, 1997, 2000.
[8] Kubale M. (Red), *Modele i metody kolorowania grafów*. Wydawnictwo Naukowo-Techniczne, Warszawa 2002.
[9] Czaderna P., Wala K., *Performance comparison of the AS and GTS algorithms for weighted maximum leaf spanning tree problem*. Prace Naukowe Politechniki Warszawskiej, Elektronika, z. 160, Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa 2007, 43–50.