Robert Janczewski, Łukasz Kuszner, Adam Nadolski
Politechnika Gdańska

ALGORYTMY ZACHŁANNE I ICH ZASTOSOWANIE W PROBLEMIE PRZYDZIAŁU CZĘSTOTLIWOŚCI

Streszczenie. Problem przydziału częstotliwości (PPCz) to następująco sformułowane zagadnienie — na pewnym obszarze znajduje się grupa nadajników radiowych; trzeba przydzielić im częstotliwości w taki sposób, by nie zakłócały się w trakcie nadawania i by wykorzystane pasmo częstotliwości miało jak najmniejszą szerokość. Zagadnienie to modeluje się zazwyczaj na gruncie teorii grafów korzystając z tzw. kontrastowych pokolorowań i *T*-rozpiętości, która jest teoriografowym odpowiednikiem szerokości przydzielonego pasma częstotliwości. Wyznaczanie *T*-rozpiętości jest silnie NP-trudne, więc w zastosowaniach stosuje się różne szybkie heurystyki. Niniejszy referat poświęcony jest jednej z takich heurystyk — algorytmowi zachłannemu.

1. Wprowadzenie

Problem przydziału częstotliwości [5] (PPCz) to następująco sformułowane zagadnienie — na pewnym obszarze znajduje się grupa nadajników radiowych, którym trzeba przydzielić częstotliwości w taki sposób, by nie zakłócały się w trakcie nadawania i by wykorzystane pasmo częstotliwości miało jak najmniejszą szerokość. Zagadnienie to modeluje się na gruncie teorii grafów przy pomocy czterech pojęć: zbiorów odległości zakazanych *T*, grafów interferencji *G*, kontrastowych pokolorowań (*T*-pokolorowań) i *T*-rozpiętości.

Wierzchołkami grafu interferencji *G* są rozważane nadajniki; zbiór wierzchołków grafu *G* oznaczać będziemy dalej literą *V*, a jego wierzchołki symbolami v_1, v_2, ..., v_n (*n* jest tutaj liczbą nadajników). Krawędzie łączą w grafie *G* te i tylko te wierzchołki-nadajniki, które mogą się zakłócać w trakcie nadawania; liczbę krawędzi grafu G oznaczać będziemy literą *m*.

Zbiór odległości zakazanych *T* to taki skończony podzbiór zbioru liczb całkowitych, że $0 \in T$. To, które liczby należą do zbioru *T* zależy od tego, jakie czynniki powodują zakłócenia w trakcie nadawania — elementy zbioru *T* powinny być tak dobrane, żeby po przydzieleniu

sąsiadującym wierzchołkom-nadajnikom częstotliwości o odległości (tj. module z różnicy) nie należącej do zbioru T nadajniki te nie zakłócały się.

Kontrastowym pokolorowaniem lub T-pokolorowaniem grafu G nazywa się każdą funkcję c, która przyporządkowuje wierzchołkom grafu G liczby całkowite (zwane dalej kolorami) w taki sposób, że $|c(v_i) - c(v_j)| \notin T$ dla każdej pary sąsiadujących w grafie G wierzchołków v_i, v_j. Tak jak kolory są odpowiednikami częstotliwości w PPCz, tak kontrastowe pokolorowania są odpowiednikami przydziałów częstotliwości. Jasne jest także, że rozpiętość pokolorowania c, oznaczana dalej $\mathrm{sp}(c)$, a definiowana jako różnica pomiędzy największym, a najmniejszym kolorem wykorzystanym przez c, jest odpowiednikiem szerokości pasma częstotliwości w PPCz. Wreszcie odpowiednikiem rozwiązania PPCz w tym modelu jest każde pokolorowanie kontrastowe, którego rozpiętość jest minimalna (takie T-pokolorowania nazywane są optymalnymi). Tę najmniejszą możliwą z rozpiętości T-pokolorowań grafu G oznacza się symbolem $\mathrm{sp}_T(G)$ i nazywa jego T-rozpiętością.

Wyznaczanie T-rozpiętości jest silnie NP-trudne nawet dla grafów podkubicznych [4, 6] (tj. grafów, w których każdy wierzchołek ma co najwyżej 3 sąsiadów) oraz pełnych [7] (grafów, w których każde dwa różne wierzchołki z sobą sąsiadują). Oznacza to, że nie znamy żadnego szybkiego (wielomianowego) algorytmu, który potrafiłby T-rozpiętość wyznaczać i co więcej, wszystko wskazuje na to, że takiego algorytmu nie ma (taki algorytm może istnieć tylko wtedy, gdy $P = NP$). Co więcej, jak wykazano w [3], szybkie algorytmy przybliżone o stałym błędzie względnym lub bezwzględnym istnieją tylko wtedy, gdy istnieje szybki algorytm optymalny. Zatem w sytuacji, w której musimy w efektywny sposób obliczać wartość T-rozpiętości, musimy skorzystać z jakiegoś algorytmu heurystycznego. Jedną z takich heurystyk jest zaadaptowany z klasycznego kolorowania grafów algorytm zachłanny. W następnych punktach omówiona zostanie zasada działania tego algorytmu, jego teoretyczne własności oraz potencjalne uogólnienia. Następnie przedstawione zostaną wyniki testów komputerowych, którym poddano algorytm zachłanny i jego zmodyfikowane wersje.

2. Algorytm zachłanny i jego własności

Algorytm zachłanny działa następująco. Na samym początku — w zerowym kroku — numeruje wierzchołki grafu G liczbami 1, 2, ..., n (istnieją różne sposoby numerowania wierzchołków prowadzące do różnych wariantów algorytmu zachłannego, takich jak T-SL, T-

LF czy też T-DSATUR; ich omówienie można znaleźć np. w [6]). Następnie — w i-tym kroku (i = 1, 2, ..., n) — nadaje wierzchołkowi v_i najmniejszy dopuszczalny kolor, tj. przyjmuje, że $c(v_i)$ jest najmniejszą nieujemną liczbą całkowitą k, dla której $|k - c(v_j)| \notin T$ dla każdego $j < i$ takiego, że v_j sąsiaduje z v_i. Dla przykładu, jeżeli G jest 3-wierzchołkowym grafem pełnym, a $T = \{0, 1, 4, 5\}$, to algorytm zachłanny pokoloruje graf G przydzielając jego wierzchołkom kolory 0, 2 i 8, uzyskując w ten sposób T-pokolorowanie o rozpiętości równej 8.

Złożoność obliczeniowa algorytmu zachłannego zależy oczywiście od tego, jaką metodą numerujemy wierzchołki i w ogólności jest równa $O(\text{num}(G)+|T|n^2)$, gdzie num(G) to koszt ponumerowania wierzchołków grafu G (w naszym przypadku, przy numerach nadawanych losowo, num(G) = $O(n)$). Wiemy także, że T-pokolorowania tworzone przez algorytm zachłanny mogą mieć rozpiętość różniącą się o dowolnie wiele od T-rozpiętości kolorowanego grafu (dla przykładu — jeżeli k jest liczbą naturalną, $T = \{0, 1, ..., 6k\} \setminus \{2k, 3k, 6k\}$, a G jest 3-wierzchołkowym grafem pełnym, to algorytm zachłanny pokoloruje graf G kolorami 0, $2k$, $8k$ uzyskując pokolorowanie o rozpiętości o $2k$ większej od T-rozpiętości, która jest równa $6k$). Algorytm zachłanny posiada także inne ciekawe własności, o których będzie mowa dalej. Zanim je sformułujemy, musimy wprowadzić trzy nowe oznaczenia — symbolem $c(G, T; i)$ oznaczać będziemy kolor, jaki algorytm zachłanny nadał wierzchołkowi T-kolorowanemu jako i-ty w trakcie kolorowania grafu G, symbolem $c(G, T)$ T-pokolorowanie grafu G wygenerowane przez algorytm zachłanny, a symbolem K_n n-wierzchołkowy graf pełny.

Twierdzenie 1. $c(G, T; i) = c(K_n, T; c(G, \{0\}; i) + 1)$.

Dowód. Przeprowadzimy indukcję względem i. Dla i = 1 dowodzona równość jest prawdziwa, bowiem z zasady działania algorytmu zachłannego wynika, że $c(G, T; 1) = 0$ niezależnie od tego, jaki jest graf G i jaki jest zbiór T. Załóżmy, że dowodzona równość jest prawdziwa dla każdego $j < i$. Wykażemy, że jest prawdziwa także dla i. W tym celu zauważmy, że z zasady działania algorytmu zachłannego wynika, że dla każdego $k < c(G, \{0\}; i)$ musi istnieć takie $j < i$, że wierzchołek v_i sąsiaduje z v_j i $c(G, \{0\}; j) = k$. Zatem v_i ma w momencie T-kolorowania sąsiadów pokolorowanych kolorami $c(K_n, T; k+1)$ dla każdego $k < c(G, \{0\}; i)$. Stąd $c(G, T; i) \geq c(K_n, T; c(G, \{0\}; i) + 1)$. Z drugiej strony, $c(K_n, T; c(G, \{0\}; i) + 1)$ jest kolorem, który może być użyty do T-pokolorowania v_i, bo jeżeli v_j sąsiaduje z v_i, to

$c(G, \{0\}; i) \neq c(G, \{0\}; j)$, skąd $|c(K_n, T; c(G, \{0\}; i) + 1) - c(K_n, T; c(G, \{0\}; j) + 1)| \notin T$. Stąd $c(G, T; i) \leq c(K_n, T; c(G, \{0\}; i) + 1)$, co dowodzi, że $c(G, T; i) = c(K_n, T; c(G, \{0\}; i) + 1)$. □

Twierdzenie to, mówiąc nieformalnie, informuje nas o tym, że algorytm zachłanny w trakcie T-kolorowania grafu ignoruje jego strukturę przy wyborze kolorów, bo używa dokładnie tych samych, których użyłby do T-pokolorowania grafu pełnego. Bezpośrednią konsekwencją tego twierdzenia jest równość $\mathrm{sp}(c(G, T)) = \mathrm{sp}(c(K_{\mathrm{sp}(c(G, T))+1}, T))$. Twierdzenie to ma także inne konsekwencje, które dowodzimy poniżej. Przypomnijmy, że symbolem $\deg(v)$ oznacza się stopień wierzchołka v, tj. liczbę sąsiadujących z nim wierzchołków.

Twierdzenie 2. $c(K_n, T; i) \leq |T| \cdot (i - 1)$

Dowód. Przeprowadzimy indukcję względem i. Dla $i = 1$ dowodzona równość jest prawdziwa, bowiem z zasady działania algorytmu zachłannego wynika, że $c(K_n, T; 1) = 0$ niezależnie od tego, jaki jest zbiór T. Załóżmy, że dowodzona równość jest prawdziwa dla każdego $j < i$. Wykażemy, że jest prawdziwa także dla i. W tym celu przyjmijmy, że $A_j = \{k > c(K_n, T; j): k - c(K_n, T; j) \in T\}$. Gdyby $c(K_n, T; i) > |T| \cdot (i - 1)$, to $\{c(v_{i-1}), ..., |T| \cdot (i - 1)\}$ zawierałoby się w $\{c(v_{i-1}), ..., c(v_i) - 1\}$. Ponieważ każde $k < c(v_{i-1})$ jest albo jednym z kolorów wykorzystanych przez algorytm zachłanny, albo należy do jednego ze zbiorów A_j, więc $(i - 1)(|T| - 1) = |A_1| + |A_2| + ... + |A_{i-1}| \geq |A_1 \cup ... \cup A_{i-1}| \geq |(A_1 \cup ... \cup A_{i-1}) \cap \{0, ..., c(v_{i-1}) - 1\}| + |(A_1 \cup ... \cup A_{i-1}) \cap \{c(v_{i-1}), ..., c(v_i) - 1\}| \geq c(v_{i-1}) - (i - 1) + |T| \cdot (i - 1) - c(v_{i-1}) + 1 = (i - 1)(|T| - 1) + 1$ — sprzeczność, która kończy dowód. □

Powyższe twierdzenie stwierdza, że podczas T-kolorowania grafów pełnych algorytm zachłanny nie może korzystać ze zbyt dużych kolorów, bo wierzchołek kolorowany jako i-ty musi otrzymać kolor nie większy niż $|T| \cdot (i - 1)$. Okazuje się, że własność ta uogólnia się także na dowolne grafy.

Twierdzenie 3. $c(G, T; i) \leq |T| \cdot \min\{\deg(v_i), i - 1\}$

Dowód. Zgodnie z twierdzeniem 1 zachodzi równość $c(G, T; i) = c(K_n, T; c(G, \{0\}; i) + 1)$, a zgodnie z twierdzeniem 2 nierówność $c(K_n, T; i) \leq |T| \cdot (i - 1)$. Stąd $c(G, T; i) \leq |T| \cdot c(G, \{0\}; i)$, co w połączeniu z oczywistą (wynikającą z tego, jak działa algorytm zachłanny) nierównością $c(G, \{0\}; i) \leq \min\{\deg(v_i), i - 1\}$ daje tezę. □

Z twierdzenia 3 wynika kilka ważnych wniosków. Po pierwsze, wynika z niego od razu, że złożoność algorytmu zachłannego jest równa $O(\text{num}(G)+|T|m)$, bowiem etap wyszukiwania koloru dla i-tego wierzchołka wymaga sprawdzenia co najwyżej $|T|\cdot\min\{\deg(v_i), i-1\}+1$ kolorów, a suma stopni wierzchołków w grafie jest równa $2m$. Po drugie, wynika z niego, że T-pokolorowanie wygenerowane przez algorytm zachłanny ma rozpiętość nie większą niż $|T|\cdot \max\{\min\{\deg(v_i), i-1\}: i \le n\}$. Po trzecie, wynika z niego poniższy wniosek, mówiący o tym, kiedy algorytm zachłanny ma szanse optymalnie T-pokolorować dany graf G. Przypomnijmy, że liczba chromatyczna grafu G, oznaczana symbolem χ, to najmniejsza liczba kolorów, przy pomocy których można pokolorować graf G.

Wniosek 4. Jeżeli $c(G, T; i)$ jest optymalnym T-pokolorowaniem grafu G, to $c(G, \{0\}; i)$ jest optymalnym $\{0\}$-pokolorowaniem grafu G i zachodzi równość $\text{sp}_T(G) = \text{sp}_T(K_\chi)$.

Dowód. Z twierdzenia 1 wynika, że zbiór kolorów wykorzystanych w trakcie kolorowania grafu G jest taki sam jak zbiór kolorów wykorzystany do pokolorowania grafu K_r, gdzie $r = \max\{c(G, \{0\}; i)+1: i \le n\}$. Oczywiście $r \ge \chi$, co wynika z definicji liczby chromatycznej. Stąd $\text{sp}_T(G) = \text{sp}(c(G, T)) = \text{sp}(c(K_r, T)) \ge \text{sp}_T(K_r)$. Z drugiej strony wiadomo [1], że $\text{sp}_T(G) \le \text{sp}_T(K_\chi)$. Stąd $r = \chi$ (bo jeżeli $s < r$, to $\text{sp}_T(K_s) < \text{sp}_T(K_r)$) i $\text{sp}_T(G) = \text{sp}_T(K_\chi)$, co było do wykazania. □

3. Uogólnione algorytmy zachłanne

Okazuje się, że zasadę, na której opiera się działanie algorytmu zachłannego można uogólnić, uzyskując w efekcie wolniejsze, lecz dokładniejsze heurystyki. Niech k będzie liczbą naturalną. Algorytm k-zachłanny działa następująco. W zerowym kroku działa tak samo, jak algorytm zachłanny, czyli numeruje wierzchołki grafu G. W pierwszym kroku pierwsze k wierzchołków koloruje tak, by powstało optymalne T-pokolorowanie podgrafu indukowanego przez te wierzchołki w grafie G (podgraf indukowany przez pewne wierzchołki składa się z tych wierzchołków i wszystkich łączących je z sobą w grafie G krawędzi). Następnie, każdą kolejną grupę k wierzchołków (ostatnia grupa może składać się z mniej niż k wierzchołków) koloruje tak, by powstało T-pokolorowanie podgrafu indukowanego przez wszystkie pokolorowane do tej pory wierzchołki i aby to T-

pokolorowanie miało najmniejszą możliwą rozpiętość. Dla przykładu, jeżeli G jest 5-wierzchołkowym grafem pełnym, a $T = \{0, 1, 4, 5\}$, to algorytm 2-zachłanny pokoloruje graf G przydzielając jego wierzchołkom kolory 0, 2 (pierwsza grupa), 8, 10 (druga grupa) i 16 (ostatnia grupa), uzyskując w ten sposób T-pokolorowanie o rozpiętości równej 16.

Nietrudno jest zauważyć, że algorytm zachłanny działa tak samo jak 1-zachłanny. Algorytmy k-zachłanne (dla $k > 1$) będą działały wolniej niż algorytm zachłanny, gdyż można bez trudu sprawdzić, że algorytm k-zachłanny działa w czasie $O(\text{num}(G) + n^{k+1}|T|^k)$. Co więcej, ten algorytm także nie musi generować optymalnych T-pokolorowań, chociaż łatwo jest zauważyć, że algorytm k-zachłanny nie myli się na grafach posiadających mniej niż $k + 1$ wierzchołków. Aby sprawdzić, czy i o ile algorytmy k-zachłanne są lepsze od zachłannego, poddano je opisanym dalej testom komputerowym.

4. Testy

Algorytmy k-zachłanne zostały poddane całej serii testów komputerowych. Z uwagi na wysoką złożoność obliczeniową, w testach wykorzystano tylko algorytm 1- i 2-zachłanny. W ramach pojedynczego testu (takich testów było dla każdego algorytmu 45) wykonywano wybrany algorytm tysiącu grafów losowych (o ustalonej liczbie wierzchołków n i ustalonym prawdopodobieństwie wylosowania krawędzi p) oraz wszystkich zbiorach T o maksymalnym elemencie mniejszym lub równym 6; następnie rozpiętości uzyskanych pokolorowań uśredniono, zaokrąglono do pierwszego miejsca po przecinku i umieszczono w poniższych tabelach.

Tabela 1

Wyniki testów dla algorytmu 1-zachłannego

n/*p*	10	20	30	40	50	60	70	80	90
10	3,0	5,0	6,7	8,4	10,0	11,8	13,7	16,3	20,0
20	6,1	9,2	12,3	15,1	18,3	21,4	25,3	30,2	37,3
30	8,2	12,8	16,9	20,9	25,2	30,1	35,5	42,6	52,7
40	10,2	15,6	20,9	26,2	31,5	38,1	45,0	54,1	67,4
50	12,2	18,8	24,7	31,1	37,8	45,2	53,7	64,8	81,4

Tabela 2

Wyniki testów dla algorytmu 2-zachłannego

n/*p*	10	20	30	40	50	60	70	80	90
10	3,1	5,1	6,9	8,7	10,4	12,3	14,3	17,0	20,8
20	6,4	9,8	12,9	16,3	19,4	22,7	26,8	31,6	38,9

30	9,0	13,7	18,2	22,6	27,2	32,2	37,9	44,9	55,0
40	11,3	17,3	22,9	28,5	34,3	40,8	48,3	57,4	70,7
50	13,1	20,3	27,1	33,8	41,1	48,8	57,6	69,2	85,3

Jak nietrudno zauważyć, algorytm zachłanny okazał się lepszy we wszystkich badanych przypadkach, przy czym różnica pomiędzy badanymi algorytmami nie jest duża (rzędu kilku procent) i rośnie wraz ze wzrostem gęstości grafu. Innymi słowy, algorytm 2-zachłanny zachowuje się najlepiej dla grafów rzadkich; im graf jest gęstszy, tym gorsze są rezultaty uzyskiwane przez algorytm 2-zachłanny. Dokładniejsza analiza (przejrzenie i porównanie wyników uzyskanych na poszczególnych grafach przez oba algorytmy, a nie tylko porównanie średniej) pokazuje jednak, że nie zawsze algorytm zachłanny generuje lepsze wyniki niż 2-zachłanny. Warto zatem zbadać, jak zachowuje się poprawiony algorytm 2-zachłanny, tzn. algorytm, który najpierw koloruje graf zachłannie, a potem 2-zachłannie i jako wynik swojego działania zwraca lepsze z uzyskanych T-pokolorowań. Wyniki testów, którym poddano ten algorytm znajdują się w poniższej tabelce. Pokazują one, że poprawiony algorytm 2-zachłanny jest o kilkanaście procent lepszy od algorytmu zachłannego dla grafów rzadkich i o kilka procent lepszy dla grafów gęstych.

Tabela 3

Wyniki testów dla poprawionego algorytmu 2-zachłannego

n/*p*	10	20	30	40	50	60	70	80	90
10	2,7	4,7	6,4	8,0	9,6	11,5	13,4	16,0	19,9
20	5,8	8,8	11,8	14,6	17,8	20,9	24,8	29,6	36,8
30	7,9	12,5	16,4	20,5	24,6	29,5	34,9	41,9	52,1
40	9,9	15,2	20,5	25,8	31,0	37,6	44,4	53,3	66,6
50	10,6	18,5	24,3	30,6	37,4	44,6	53	64,1	80,5

5. Podsumowania

Podsumowując, algorytm zachłanny jest heurystyką, która posiada ciekawe własności teoretyczne, z których najważniejszą jest to, że generowane $\underline{T}$-pokolorowania nie mogą mieć zbyt dużej rozpiętości (dla przykładu dla grafów podkubicznych rozpiętość nie przekracza $2|T|$, co oznacza, że dla tych grafów algorytm zachłanny jest algorytmem o stałym błędzie względnym!). Co więcej, niewielka modyfikacja zasady działania algorytmu zachłannego pozwala poprawić jego zachowanie o kilka do kilkunastu procent (kosztem oczywiście czasu

działania — poprawiony algorytm zachłanny jest znacznie wolniejszy od nie zmodyfikowanego).

LITERATURA:

1. Cozzens M.B., Roberts F.S.: T-colorings of graphs and the channel assignment problem. Congressus Numerantium 35 (1982), pp. 191-208.
2. Cozzens M.B., Roberts F.S.: Meaningfulness of conclusions about greedy algorithms for *T*-coloring. Rutgers University, Dept. of Mathematics, Technical Report 47/90.
3. Giaro K.: Złożoność obliczeniowa problemu *T*-kolorowania grafu w zależności od jego maksymalnego stopnia. Politechnika Gdańska, Wydział ETI, Raport Techniczny 34/99.
4. Giaro K., Janczewski R., Małafiejski M.: The complexity of *T*-coloring problem for graphs with small degree. Discrete Applied Mathematics 129 (2003), pp. 361-369.
5. Hale W.K.: Frequency assignment. Theory and applications. Proceedings of IEEE, vol. 68, no. 12 (1980), pp. 1497-1514.
6. Janczewski R.: Kontrastowe kolorowanie grafów i jego zastosowania. Rozprawa doktorska, Politechnika Gdańska, Wydział ETI, Gdańsk 2001.
7. Jansen K.: A rainbow about *T*-colorings for complete graphs. Discrete Mathematics 154 (1996), pp. 129-139.
8. Tesman B.A.: *T*-colorings, list *T*-colorings and set list *T*-colorings of graphs. Ph. D. Thesis, Rutgers University, New Brunswick, New Jersey 1989.
9. Tesman B.A.: List *T*-colorings of graphs. Discrete Applied Mathematics 45 (1993), pp. 277-289.

Recenzent:

GREEDY ALGORITHMS AND THEIR APPLICATION TO THE FREQUENCY ASSIGNMENT PROBLEM

Abstract:

Frequency assignment problem (FAP) can be defined as follows. There are several radio transmitters situated in a certain region of a plane; a channel is to be assigned to each of them in such a way that there is no interference during transmitting and the span of used frequency band is minimal. The FAP problem can be represented as one of graph coloring

problems. More precisely, it can be represented as the problem of computing the *T*-span of the so-called interference graph. The problem of computing the *T*-span is NP-hard in the strong sense so different heuristics are used in applications. The paper is devoted to the greedy algorithm which is one of these heuristics.