ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 2006
Seria: AUTOMATYKA z. 143 Nr kol. 1726

Dariusz DERENIOWSKI
Politechnika Gdańska

GENEROWANIE SĄSIEDZTWA W ALGORYTMACH LOKALNYCH POSZUKIWAŃ UPORZĄDKOWANEGO KOLOROWANIA GRAFÓW

Streszczenie. Przedstawienie rozwiązań problemów kombinatorycznych w postaci permutacji daje podstawy do konstrukcji algorytmów lokalnych poszukiwań. Uporządkowane pokolorowanie grafu można zapisać w postaci permutacji wierzchołków grafu. Podstawowe operacje, prowadzące do generowania sąsiedztwa rozwiązania, to zamiana dwóch elementów lub przesunięcie elementu permutacji. W artykule wskazujemy metodę, pozwalającą na wykonanie takich operacji w czasie $O(m)$, przy założeniu, że dane jest drzewo eliminacji wyjściowej permutacji.

NEIGHBORHOOD GENERATION IN LOCAL SEARCH ALGORITHMS FOR VERTEX RANKING OF GRAPHS

Summary. Representing solutions to combinatorial problems as permutations allows us to use local search algorithms for solving them. A vertex ranking of a graph can be represented by a permutation of the vertices of the graph. Basic operations for generating a neighborhood of a current solution are swapping two elements of the permutation or changing a position of an element. We show how to perform such operations in time $O(m)$ assuming that an elimination tree of the current permutation is given.

1. Wprowadzenie

Uporządkowane k-pokolorowanie wierzchołków grafu prostego $G = (V, E)$ to funkcja $c\colon V \to \{1, \ldots, k\}$ taka, że każda ścieżka łącząca dwa wierzchołki u, v o jednakowym kolorze (tzn. $c(u) = c(v)$) zawiera wierzchołek w taki, że $c(w) > c(u)$. Najmniejsza liczba k, dla której istnieje uporządkowane k-pokolorowanie, to *uporządkowana liczba chromatyczna* grafu G, oznaczana symbolem $\chi_r(G)$. Mówimy, że c jest *optymalne* jeśli używa $\chi_r(G)$ kolorów.

Problem uporządkowanego kolorowania grafów jest obliczeniowo trudny w ogólności [7] oraz dla grafów dwudzielnych [1] i cięciwowych [4]. Z drugiej stro-

ny, istnieje szereg klas grafów, dla których optymalne rozwiązanie można znaleźć w wielomianowym czasie: drzewa [8], grafy przedziałowe, permutacyjne, grafy łuków (ang. circular-arc) [3]. Dla grafów ogólnych znany jest wielomianowy algorytm przybliżony o funkcji dobroci $O(\log^2 n)$, gdzie n jest liczbą wierzchołków grafu [2]. Uporządkowane kolorowanie wierzchołków grafu znajduje zastosowanie przy równoległej faktoryzacji symetrycznych i dodatnio określonych macierzy metodą Choleskiego [5, 6].

Problem wyznaczania optymalnego uporządkowanego pokolorowania jest równoważny problemowi szukania drzewa eliminacji o minimalnej wysokości. W następnym rozdziale zostanie podana definicja drugiego z problemów oraz zostaną wskazane pewne związki łączące oba problemy. Podstawowym faktem, który będziemy wykorzystywać, jest to, że każdemu drzewu eliminacji odpowiada pewna permutacja wierzchołków grafu. To stwarza podstawy do konstrukcji metaheurystyk lokalnych poszukiwań, które operują wyłącznie na permutacji wierzchołków grafu. Funkcją kosztu dla danej permutacji jest wysokość odpowiedniego drzewa eliminacji (lub równoważnie liczba kolorów w odpowiednim uporządkowanym pokolorowaniu). Rozdział 3. podaje twierdzenia, które pozwalają dokonać pewnych modyfikacji danej permutacji i obliczyć zmianę funkcji kosztu w czasie $O(m)$, gdzie m jest liczbą krawędzi grafu G. Obliczenie drzewa eliminacji dla premutacji na podstawie grafu G wymaga czasu $O(m \log n)$, gdzie n jest liczbą wierzchołków grafu [6]. Rozdział 4. zawiera krótki opis wykorzystania podanych twierdzeń do konstrukcji metaheurystyk lokalnych poszukiwań.

2. Drzewa eliminacji

Niech będzie dany graf prosty G oraz pewna permutacja jego wierzchołków $p = (p(1), \ldots, p(n))$. Przyjmijmy, że $G_p(0) = G$. *Eliminacja wierzchołka* $p(i)$ polega na dodaniu do grafu $G_p(i-1)$ krawędzi w taki sposób, aby wszyscy sąsiedzi $p(i)$, należący do zbioru $\{p(i+1), \ldots, p(n)\}$, tworzyli podgraf pełny w $G_p(i-1)$. Nowo utworzony graf to $G_p(i)$. Dla skrócenia zapisu oznaczamy $G_p(n) = G_p$.

Definicja 1. Drzewem eliminacji *nazywamy zakorzenione drzewo* T_p, *w którym wierzchołek* $p(i)$ *jest ojcem wierzchołka* $p(j)$, *o ile*

1. $i > j$,
2. $\{p(i), p(j)\} \in E(G_p)$,
3. $\{p(i), p(k)\} \notin E(G_p)$, *dla* $k = i+1, \ldots, j-1$.

Przez $h(T_p)$ *oznaczamy wysokość drzewa* T_p, *którą definiujemy jako długość najdłuższej ścieżki (długość ścieżki definiujemy jako liczbę jej krawędzi) łączącej korzeń z liściem. Symbol* $h(G)$ *oznacza wysokość najniższego drzewa eliminacji.*

Poniższe twierdzenie wskazuje na równoważność problemów szukania optymalnego uporządkowanego pokolorowania oraz drzewa eliminacji o minimalnej wysokości.

Twierdzenie 1. ([3]) *Dla danego grafu prostego G zachodzi $h(G)+1=\chi_r(G)$.* □

Co więcej, dysponując drzewem eliminacji T_p, możemy utworzyć uporządkowane $(h(T_p)+1)$-pokolorowanie w taki sposób, iż wierzchołki należące do i-tego poziomu w drzewie otrzymują kolor i (korzeń drzewa należy do poziomu numer $h(T_p)+1$ oraz jeśli pewien wierzchołek należy do poziomu numer i, to wszyscy jego synowie należą do poziomu $i+1$).

3. Podstawowa operacja modyfikacji permutacji

W dalszej części będziemy potrzebować kilku pojęć z teorii grafów. Niech będzie dany graf prosty $G=(V(G),E(G))$. Często piszemy V lub E zamiast $V(G)$ lub $E(G)$. Graf jest *spójny* jeśli każda para jego wierzchołków jest połączona ścieżką. Jeśli $S\subseteq V$, to $G-S=(V\setminus S,\{\{u,v\}\in E : u,v\notin S\})$. Zbiór S jest *separatorem* grafu G jeśli $G-S$ nie jest spójny. Każdy maksymalny spójny podgraf G jest nazywany jego *składową spójności*.

Dla danego drzewa eliminacji T_p oraz dowolnego wierzchołka $v\in V$ definiujemy $T_p[v]$ jako poddrzewo T_p indukowane przez wszystkich potomków wierzchołka v (wraz z v) w drzewie T_p. Ponadto niech $T_p-v=T_p-V(T_p[v])$.

W niniejszym rozdziale zakładamy, że permutacja $p\colon V\to\{1,\ldots,n\}$ jest dana, gdzie $n=|V|$. Definiujemy permutację p' następująco:

$$p'=(p(1),\ldots,p(i-1),p(j),p(i),\ldots,p(n)). \tag{1}$$

Innymi słowy, p' powstaje z p poprzez przesunięcie wierzchołka $p(j)$ na i-tą pozycję. Naszym celem jest obliczenie drzewa eliminacji $T_{p'}$ przy założeniu, że T_p jest dane.

Zauważmy, że p' można otrzymać z p poprzez dokonanie $O(|i-j|)$ zamian sąsiednich elementów. Zakładamy dalej, że

$$p'=(p(1),\ldots,p(i-1),p(i+1),p(i),p(i+2),\ldots,p(n)). \tag{2}$$

Lemat 1. *Jeśli $p(i)$ oraz $p(i+1)$ nie są spokrewnione w T_p, to $T_p=T_{p'}$.*

Dowód. Zauważmy, że $G_p(i-1)=G_{p'}(i-1)$. Z faktu, że wierzchołki $p(i)$ oraz $p(i+1)$ nie są spokrewnione w T_p wynika, że nie są one sąsiednie w $G_p(i-1)$, gdyż w przeciwnym wypadku byłyby spokrewnione w G_p i na mocy definicji mielibyśmy, że $p(i+1)$ jest ojcem $p(i)$ w T_p. Niech $X_i,X_{i+1}\subseteq\{p(i+2),\ldots,p(n)\}$ oznaczają sąsiadów wierzchołków $p(i),p(i+1)$ w grafie $G_p(i-1)$. Eliminacja $p(i)$ prowadzi do utworzenia grafu $G_p(i)$, w którym sąsiedztwo (uwzględniamy ponownie tylko wierzchołki, które nie zostały jeszcze wyeliminowane) $p(i+1)$ to zbiór X_{i+1}. Podobnie sąsiedztwem $p(i)$ w $G_{p'}(i-1)$ będzie X_i. Oznacza to, że eliminując $p(i),p(i+1)$ w dowolnej kolejności, utworzymy w obu przypadkach podgrafy pełne złożone z wierzchołków X_i i X_{i+1}, więc $G_p(i+1)=G_{p'}(i+1)$. □

Rozważamy zatem w dalszej części sytuację, gdy $p(i), p(i+1)$ są spokrewnione w T_p. Wprost z definicji drzewa eliminacji wynika, że w takim przypadku $p(i+1)$ jest ojcem $p(i)$.

Lemat 2. ([6]) *Dla dowolnego $v \in V$, wierzchołki poddrzewa $T_p[v]$ indukują spójny podgraf w G.* □

Lemat 3. ([6]) *Wierzchołki $p(i)$ oraz $p(j)$ są sąsiednie w G_p wtedy i tylko wtedy, gdy są one połączone w G ścieżką zawierającą wierzchołki należące do zbioru $\{p(1), \ldots, p(\min\{i,j\} - 1)\}$.* □

Procedurę obliczającą $T_{p'}$ można opisać następująco.

1. ojcem wierzchołka $p(i)$ w $T_{p'}$ jest ojciec $p(i+1)$ w T_p;
2. ojcem wierzchołka $p(i+1)$ w $T_{p'}$ jest $p(i)$;
3. ojcem $p(l)$ w $T_{p'}$ jest $p(i+1)$ dla każdego potomka $p(l)$ wierzchołka $p(i)$ w T_p takiego, że $p(i+1)$ jest sąsiedni w G do pewnego wierzchołka składowej spójności indukowanej przez wierzchołki poddrzewa $T_p[p(l)]$;
4. dla pozostałych wierzchołków $p(l)$, ojciec w $T_{p'}$ jest taki, jak ojciec $p(l)$ w T_p;

Lemat 4. *Powyższa procedura wyznacza drzewo eliminacji $T_{p'}$ dla permutacji p' danej wzorem* (2).

Dowód. Dowodzimy, że ojcem (oznaczmy go przez $p(t)$) wierzchołka $p(i)$ w $T_{p'}$ jest ojciec $p(i+1)$ w T_p. Z lematu 3 wynika, że istnieją ścieżki w G, łączące odpowiednio $p(i)$ z $p(i+1)$ oraz $p(i+1)$ z $p(t)$, obie złożone z wierzchołków ze zbioru $\{p(1), \ldots, p(i)\}$. Oznacza to istnienie odpowiedniej ścieżki pomiędzy $p(i)$ oraz $p(t)$, więc $\{p(i), p(t)\} \in E(G_{p'})$. Podobnie można pokazać, że $\{p(i), p(t')\} \notin E(G_p)$, gdzie $i+1 < t' < t$, co oznacza, że $p(t')$ nie jest spokrewniony z $p(i+1)$ w T_p.

Dowodzimy, że ojcem wierzchołka $p(i+1)$ w $T_{p'}$ jest $p(i)$. Z definicji drzewa eliminacji wynika, że wystarczy pokazać, że $\{p(i), p(i+1)\} \in E(G_p)$. Teza wynika z lematu 3, gdyż $\{p(1), \ldots, p(i-1)\} = \{p'(1), \ldots, p'(i-1)\}$.

Niech $p(l)$ będzie dowolnym potomkiem $p(i)$ w T_p. Niech $G' \subseteq G$ będzie spójnym podgrafem indukowanym przez wierzchołki poddrzewa $T_p[p(l)]$. Na mocy lematu 2 taki podgraf istnieje. Z definicji drzewa eliminacji wynika, że $p(l)$ jest wierzchołkiem o najwyższym indeksie w permutacji p, spośród wierzchołków należących do $T_p[p(l)]$. Jeśli $p(i+1)$ jest sąsiedni do pewnego wierzchołka należącego do $T_p[p(l)]$, to w G istnieje ścieżka łącząca $p(l)$ z $p(i+1)$, zawierająca wierzchołki należące do $T_p[p(l)]$. Na mocy lematu 3 mamy, że $p(l)$ jest synem $p(i+1)$ w $T_{p'}$.

Jeśli $p(l)$ jest potomkiem $p(i)$ w T_p, to $T_p[p(l)] = T_{p'}[p(l)]$. Jeśli $p(l)$ nie jest spokrewniony z $p(i)$ lub $p(i+1)$ w T_p, to $T_p[p(l)] = T_{p'}[p(l)]$. Jeśli $p(l)$ jest przodkiem $p(i+1)$ w T_p (lecz nie ojcem), to z definicji mamy, że zbiory potomków wierzchołka $p(l)$ w drzewach T_p i $T_{p'}$ są równe. □

Twierdzenie 2. *Niech będzie dana permutacja p oraz odpowiadające jej drzewo eliminacji T_p. Drzewo eliminacji $T_{p'}$, gdzie p' jest dana wzorem* (1) *można wyznaczyć w czasie $O(m)$.*

Dowód. Kroki 1 i 2 powyższego algorytmu mogą zostać zrealizowane w stałym czasie. Czas realizacji kroku 3 jest proporcjonalny do liczby krawędzi podgrafu G indukowanego przez wierzchołki w $T_p[l]$. Dla każdego wierzchołka x obliczenie sąsiedztwa wierzchołków w $T_p[x]$ w grafie G wystarczy obliczyć co najwyżej jednokrotnie. □

4. Podsumowanie

Twierdzenie 2, podane w poprzednim podrozdziale, można wykorzystać do wydajniejszej implementacji metaheurystyk lokalnych poszukiwań. Algorytm, w którym sąsiedztwo danej permutacji to zbiór wszystkich permutacji możliwych do otrzymania poprzez przesunięcie wybranego wierzchołka $p(i)$ w dowolne miejsce, ma złożoność $O(In^2m)$, gdzie I jest liczbą iteracji głównej pętli algorytmu, w której następuje wybór najlepszego sąsiedniego rozwiązania. Zamiana dwóch wierzchołków $p(i), p(j)$ miejscami może być zaimplementowana jako złożenie dwóch przesunięć, więc taka operacja prowadzi do algorytmu o złożoności podobnej do wymienionego powyżej. Zauważmy, że Lemat 1 pozwala na uniknięcie obliczeń w przypadku wielu zamian elementów, jednak w pesymistycznym przypadku złożoność algorytmu jest taka jak podano powyżej.

LITERATURA

1. Bodlaender H., Deogun J.S., Jansen K., Kloks T., Kratsch D., Müller H., Tuza Z.: Rankings of graphs, SIAM J. Discrete Math. **11**, 1998, p. 168–181.
2. Bodlaender H., Gilbert R.J., Hafsteinson H., Kloks T.: Approximating treewidth, pathwidth, frontsize, and shortest elimination tree, J. Algorithms **18**, 1995, p. 238–255.
3. Deogun J.S., Kloks T., Kratsch D., Müller H.: On the vertex ranking problem for trapezoid, circular-arc and other graphs, Discrete Appl. Math. **98**, 1999, p. 39–63.
4. Dereniowski D., Nadolski A.: Vertex rankings of chordal graphs and weighted trees, Inform. Process. Letters **98**, 2006, p. 96–100.
5. Liu J.W.H.: Modification of the minimum degree algorithm by multiple elimination, ACM Transactions on Mathematical Software **12**, 1985, p. 615–627.
6. Liu J.W.H.: The role of elimination trees in sparse factorization, SIAM J. Matrix Analysis and Appl. **11**, 1990, p. 134–172.
7. Pothen A.: The complexity of optimal elimination trees, Tech. Report **CS-88-13**, The Pennsylvania State University, 1988.

8. Schäffer A.A.: Optimal node ranking of trees in linear time, Inform. Process. Letters **33**, 1989/90, p. 91–96.

Recenzent: Prof. dr hab. inż. Adam Janiak

Abstract

Given a simple graph G, a function c is a vertex k-ranking of G if it uses k colors and each path connecting two vertices with the same colors contains a vertex with a bigger color. The smallest number k for which there exists a vertex k-ranking is called a vertex ranking number of G and is denoted by $\chi_r(G)$. The vertex ranking problem is equivalent to the problem of finding a minimum height elimination tree of G. This implies that each vertex ranking of G can be represented as a permutation of the vertices of G, which allows us to use local search algorithms. In this paper we give an algorithm which for given permutation p and its elimination tree T_p finds in time $O(m)$ an elimination tree $T_{p'}$ where p' is a permutation obtained from p by changing the position of a selected element. This algorithm can be used to improve the efficiency of some local search heuristics.