ZESZYTY NAUKOWE POLITECHNIKI ŚLĄSKIEJ 1998
Seria: AUTOMATYKA z. 123 Nr kol. 1389

Piotr BOROWIECKI
Zakład Matematyki Dyskretnej i Informatyki Politechniki Zielonogórskiej

KOLOROWANIE GRAFÓW W TRYBIE ON-LINE

Streszczenie. Natura wielu rzeczywistych problemów powoduje, że muszą być one rozwiązywane przy użyciu algorytmów działających w trybie on-line. Algorytm rozwiązujący taki problem w trybie off-line ma wtedy poważnie ograniczoną przydatność praktyczną. Niniejsza praca poświęcona jest problemowi kolorowania grafów w trybie on-line. Przedstawione są najważniejsze wyniki dotyczące tego problemu, w tym przegląd oszacowań liczby on-line chromatycznej, przykłady zastosowań oraz opis podstawowych algorytmów kolorowania grafów w trybie on-line.

ON-LINE GRAPH COLORING

Summary. It is an inherent character of many problems, that they have to be solved on-line. Any off-line algorithm has a limited power to solve such problems in practice. In this paper we investigate the problem of on-line graph coloring. We review most important results and bounds on the on-line chromatic number. An application examples and description of basic on-line algorithms for graph coloring are included.

1. Problem kolorowania grafów

Problem kolorowania grafów występuje najczęściej w jednej z trzech klasycznych postaci: kolorowanie wierzchołków grafu, kolorowanie krawędzi grafu, równoczesne kolorowanie wierzchołków i krawędzi grafu. Wyniki opisywane w tej pracy dotyczą kolorowania wierzchołków grafu. Będziemy rozważać grafy bez pętli i wielokrotnych krawędzi. Dla danego grafu $G = (V, E)$, zbiór V o mocy $|V| = n$ nazywamy *zbiorem wierzchołków*, a zbiór E o mocy $|E| = m$ nazywamy *zbiorem krawędzi*. Definicje wszystkich pojęć z zakresu teorii grafów, niezdefiniowane w tej pracy można znaleźć w [9]. *Kolorowanie wierzchołków grafu* $G = (V, E)$ polega na przypisaniu każdemu wierzchołkowi $v_i \in V, i = 1, \ldots, n$ takiego koloru $c(v_i)$, aby dowolne dwa sąsiednie wierzchołki v_i, v_j miały różne kolory $c(v_i) \neq c(v_j), i \neq j$. Kolorowanie wierzchołków grafu przy użyciu k kolorów nazywamy *k-pokolorowaniem*. Dowolne k-pokolorowanie wierzchołków grafu G tworzy rozbicie

$(V_1, V_2, \ldots, V_k)$ zbioru wierzchołków $V(G)$ na k podzbiorów niezależnych nazywanych *klasami kolorów*. Najmniejszą liczbę k, dla której istnieje k-pokolorowanie grafu G, nazywamy *liczbą chromatyczną* grafu G i oznaczamy $\chi(G)$.

W pracy tej przyjmiemy jedno z możliwych sformułowań problemu kolorowania grafów w postaci następującego problemu minimalizacyjnego: *Wyznaczyć rozbicie zbioru wierzchołków na najmniejszą liczbę podzbiorów niezależnych.* Problem ten, jako jeden z najtrudniejszych problemów optymalizacyjnych oraz centralny problem matematyki dyskretnej, doczekał się wraz z całą rodziną związanych z nim problemów pokrewnych bardzo bogatej literatury. Jedną z najbardziej znanych pozycji jest książka Jensena i Tofta [19] będąca unikalnym katalogiem ponad dwustu problemów ze świata teorii kolorowania grafów. Zawiera ona sformułowania poszczególnych problemów, przegląd kluczowych dla nich rezultatów, pytań i otwartych problemów wytyczających kierunki dla dalszych badań.

2. Kolorowanie grafów w trybie on-line a kolorowanie w trybie off-line

Algorytm rozwiązujący dany problem w trybie on-line, krótko *algorytm on-line*, otrzymuje dane wejściowe w postaci sekwencji żądań, obsługując każde żądanie natychmiast po jego otrzymaniu, a obsługę następnego podejmuje po zakończeniu obsługi żądania poprzedniego. Zakłada się, że obsługa żądania przebiega bez znajomości przyszłych żądań oraz że raz obsłużone żądanie nie może zostać obsłużone ponownie. Zatem obsługa każdego z żądań prowadzi do wygenerowania części końcowego rozwiązania problemu. Algorytmy rozwiązywania problemów w trybie on-line porównuje się z *algorytmami off-line*, które z góry otrzymują pełną sekwencję żądań. Od algorytmu off-line również wymaga się obsłużenia każdego żądania, ale wybór sposobu obsługi może być oparty na znajomości całej sekwencji. Zatem, algorytm off-line "zna przyszłość", natomiast algorytm on-line nie.

Stosując powyższy opis do problemu kolorowania grafów, przyjmujemy, że w trybie off-line algorytm ma daną strukturę całego grafu. W trybie on-line struktura kolorowanego grafu nie jest znana z góry. Kolejne wierzchołki grafu G prezentowane są na wejściu algorytmu w niezależnym od algorytmu porządku $(v_1, v_2, \ldots, v_n)$. W momencie prezentacji wierzchołka v_i odsłaniane są także wszystkie istniejące krawędzie łączące v_i z zaprezentowanymi wcześniej wierzchołkami $v_1, \ldots, v_{i-1}$. Algorytm nieodwracalnie przyporządkowuje prezentowanemu wierzchołkowi kolor $c(v_i)$ przed prezentacją następnego wierzchołka. Kolorowanie grafu G w trybie on-line można zinterpretować jako grę na grafie G dwóch

przeciwników, prezentera i malarza. Celem malarza jest użycie jak najmniejszej liczby kolorów, czemu przeciwstawia się strategię prezentera, szukającego takiego uporządkowania wierzchołków grafu, które zmusi malarza do użycia więcej niż $\chi(G)$ kolorów. Graf G wraz z zadanym uporządkowaniem wierzchołków nazywamy *on-line prezentacją grafu G*. *Liczba on-line chromatyczna grafu G dla algorytmu A* kolorowania grafów w trybie on-line, oznaczana $\chi_A(G)$, zdefiniowana jest jako maksymalna liczba kolorów użytych do pokolorowania grafu G wśród pokolorowań tego grafu wygenerowanych przez algorytm A, dla wszystkich możliwych on-line prezentacji grafu G. Z przytoczonych powyżej definicji dla dowolnego algorytmu A i dowolnego grafu G mamy

$$\chi(G) \leq \chi_A(G).$$

Nietrudno zauważyć, że liczba klas kolorów w otrzymanym w trybie on-line pokolorowaniu w istotny sposób zależy od przyjętego uporządkowania wierzchołków, jednak wyznaczanie najlepszego pokolorowania przez przegląd wszystkich $n!$ możliwych uporządkowań jest w praktyce nie do przyjęcia, nawet dla niedużych wartości n. Stąd wysiłki autorów wielu prac skierowane zostały w stronę badania różnych uporządkowań i charakteryzacji tych, które prowadzą do lepszych rozwiązań [29, 10, 3, 4]. Jednak wspomniane prace dają w rezultacie algorytmy kolorowania off-line, w których działaniu wyróżnić można najczęściej dwie fazy: (1) wyznaczenie dla danego z góry grafu uporządkowania wierzchołków oraz (2) kolorowanie wierzchołków zgodnie z wyznaczonym porządkiem. Przegląd znanych algorytmów, ich zastosowań oraz bogatą bibliografię znaleźć można w przeglądowych pracach Kubale [25, 26] traktujących o kolorowaniu w trybie off-line zarówno wierzchołków, jak i krawędzi grafów. W dalszej części tej pracy zajmiemy się algorytmami kolorowania wierzchołków grafu w trybie on-line.

3. Algorytmy kolorowania grafów w trybie on-line

Definicja optymalnego algorytmu off-line nie sprawia większych trudności; algorytm taki dla dowolnych danych wejściowych wyznacza *rozwiązanie dokładne* będące ekstremum pewnej funkcji kryterialnej. Definicja podobnego pojęcia dla algorytmu on-line przez dłuższy czas sprawiała trudności, stąd dla wielu problemów ich "wersje on-line" długo nie były badane. *Efektywność* algorytmu on-line rozumiana będzie jako cecha gwarantująca otrzymanie rozwiązania o przewidywalnej jakości, nie odbiegającej znacząco

od rozwiązania dokładnego. Bardziej formalnie, algorytm A kolorowania on-line nazywamy efektywnym dla rodziny grafów $\mathcal{G}$, jeżeli istnieje funkcja $f(\chi)$ taka, że liczba kolorów użytych przez A do pokolorowania grafu $G \in \mathcal{G}$ w jego dowolnej on-line prezentacji wynosi co najwyżej $f(\chi(G))$. W wielu przypadkach możliwe jest podanie oszacowań przez funkcję $f(\omega)$, gdzie $\omega = \omega(G)$ jest liczbą wierzchołków w największym podgrafie pełnym (największej klice) grafu G. Dla dowolnego grafu G zachodzi $\chi(G) \geq \omega(G)$.

Ze względu na brak wielomianowych algorytmów kolorujących dowolne grafy za pomocą $\chi(G)$ kolorów, prace prowadzone są między innymi w kierunku konstruowania algorytmów przybliżonych (aproksymacyjnych).

Jakość rozwiązań, generowanych przez przybliżony algorytm on-line A dla grafu G, oceniać będziemy w oparciu o współczynnik $r_A(G)$ nazywany *dobrocią* algorytmu A dla grafu G i zdefiniowany dla problemu kolorowania grafów w trybie on-line jako $\chi_A(G)/\chi(G)$. W celu oceny efektywności można dla każdego algorytmu on-line A zdefiniować także *funkcję dobroci* $\rho_A : N \to R$

$$\rho_A(n) = \max r_A(G),$$

gdzie N, R oznaczają odpowiednio zbiór liczb naturalnych i rzeczywistych, a maksimum dotyczy wszystkich grafów o n wierzchołkach. Funkcja ta mówi, jak bardzo rozwiązania generowane przez algorytm A różnią się od rozwiązania dokładnego, gdy na wejściu algorytm otrzymuje "najgorsze dane". Dla kolorowania grafów w trybie on-line są to te uporządkowania wierzchołków grafu, które wymuszają użycie przez algorytm największej liczby kolorów. Łatwo zauważyć, że dla dowolnego algorytmu A kolorowania grafów mamy $1 \leq \rho_A(n) \leq n$. Przy czym $\rho_A(n) = 1$ dla każdego algorytmu dokładnego.

Pierwsze dolne oszacowanie funkcji dobroci dla algorytmu kolorowania on-line podane zostało przez Beana [2], który udowodnił, że dla dowolnego algorytmu A kolorowania on-line istnieje drzewo o n wierzchołkach, którego pokolorowanie wymaga użycia przynajmniej $1+\log_2 n$ kolorów. Zatem $\rho_A(n) > (\log_2 n)/2$. Jak podaje Lovász i in. [28], Szegedy udowodnił, że

$$\rho_A(n) \geq \frac{n}{(\log_2 n)^2}.$$

3.1. Algorytm *LST*

Nazwa algorytmu *LST* pochodzi od pierwszych liter nazwisk jego autorów Lovásza, Saksa i Trottera, którzy jako pierwsi sformułowali algorytm kolorowania dowolnych grafów w trybie on-line [28] posiadający subliniową funkcję dobroci

$$\rho_{LST}(n) \leq (1 + o(1))\frac{2n}{\log_2^* n},$$

gdzie $\log^* n$ oznacza najmniejsze k, dla którego k-krotnie iterowany logarytm $\log^{(k)} n = \log\log\ldots\log n$ (k-razy) osiąga wartość nie większą niż 1. Algorytm LST jest najlepszym z podanych do tej pory deterministycznych algorytmów on-line kolorowania dowolnych grafów. Wysoki stopień komplikacji algorytmu LST skłania do odesłania Czytelnika do jego dokładnego opisu w pracy źródłowej [28].

3.2. Algorytm zachłanny

Algorytmy zachłanne (ang. greedy) są algorytmami iteracyjnymi, których główną cechą jest podejmowanie, w każdej z iteracji, decyzji optymalnej z punktu widzenia aktualnej iteracji (decyzji lokalnie optymalnej), lecz prowadzące do rozwiązań końcowych, które nie muszą być optymalne globalnie. Algorytm zachłanny FF (ang. First-Fit), zastosowany do kolorowania wierzchołków grafu G w trybie on-line, realizuje strategię zachłanną, przyporządkowując każdemu wierzchołkowi możliwie najmniejszy kolor. W rozbiciu $(V_1, V_2, \ldots, V_k)$ na k klas kolorów, otrzymanym przy użyciu algorytmu FF, każdy zbiór niezależny V_i jest maksymalny, względem relacji inkluzji, w podgrafie indukowanym przez $V_i \cup \ldots \cup V_k$. Wynikają stąd następujące własności.

Własność 1. Każdy wierzchołek $v \in V_i, i \geq 2$ ma sąsiadów we wszystkich klasach $V_j, j < i$.

Własność 2. Dla dowolnego wierzchołka $v \in V(G), deg(v) \geq c(v) - 1$.

Własność 3. W grafie G istnieje droga $P_k = v_1 v_2 \ldots v_k$ taka, że $c(v_i) = i$.

Łatwo zauważyć, że każde rozwiązanie wygenerowane przez algorytm FF kolorowania działający w trybie on-line może być wygenerowane także przez bliźniaczy algorytm działający w trybie off-line używający tej samej strategii kolorowania wierzchołków danego z góry grafu, przetwarzając je w tej samej kolejności co algorytm w trybie on-line.

4. Oszacowania liczby on-line chromatycznej

Jakość rozwiązań generowanych przez algorytmy kolorowania grafów w trybie on-line zależy w istotny sposób od własności kolorowanych grafów, stąd w wielu pracach, w nadziei na otrzymanie lepszych rozwiązań, badane jest działanie algorytmów kolorowania on-line dla wybranych klas grafów.

Ze względu na wiele zastosowań (szerzej omówionych w dalszej części pracy) szczególnym zainteresowaniem cieszy się klasa grafów przedziałów (ang. interval graphs). Graf G jest grafem przedziałów, gdy istnieje rodzina $\{I_v : v \in V(G)\}$ przedziałów liczb na

osi rzeczywistej takich, że $I_u \cap I_v \neq \emptyset$ wtedy i tylko wtedy, gdy $uv \in E(G)$. Woodall [35] i niezależnie Chrobak i Ślusarek [7] wykazali dla algorytmu FF, że jeżeli G jest grafem przedziałów, to $\chi_{FF}(G)$ jest ograniczona z góry przez funkcję kwadratową liczby $\chi(G)$. Sformułowali równocześnie problem istnienia ograniczenia liniowego. Jak podają Gyárfás i Lehel [14], Just podał oszacowanie $\chi_{FF}(G) \leq c\chi(G)\log\chi(G)$, gdzie c jest dodatnią stałą. W pracy [15] Gyárfás i Lehel wykazali, że w specjalnym przypadku $\chi_{FF}(G) \leq 24\chi(G)$. Niemal w tym samym czasie Kierstead w [20] potwierdza istnienie liniowego ograniczenia dowodząc, $\chi_{FF}(G) \leq 40\chi(G)$ dla dowolnego grafu przedziałów G. Najlepsze obecnie ograniczenie $\chi_{FF}(G) \leq 25.72\chi(G)$ podali Kierstead i Qin [22]. Historia dolnych ograniczeń dla algorytmu FF na grafach przedziałów to prace takich autorów, jak Woodall, Witsenhausen [34] oraz Chrobak i Ślusarek [8]. Najlepsze dolne ograniczenie $\chi_{FF}(G) \geq 4.45\chi(G)$ należy aktualnie do Ślusarka [31], który w tej samej pracy sformułował otwartą do dzisiaj hipotezę, że $r_{FF} = 4.5$.

Poszukując najlepszego algorytmu kolorowania on-line dla grafów przedziałów napotykamy pracę [23], w której Kierstead i Trotter wykazują istnienie algorytmu A, dla którego $\chi_A(G) \leq 3\chi(G)-2$. Co więcej, autorzy podają konstrukcję instancji problemu, dla której $\chi_A(G) = 3\chi(G)-2$, zatem ograniczenie jest najlepszym z możliwych.

Efektywność algorytmów kolorowania on-line zależna jest w dużym stopniu od obecności w kolorowanych grafach pewnych podgrafów indukowanych. Mówimy, że graf G jest *F-wolny*, jeżeli G nie zawiera *zabronionego podgrafu indukowanego* izomorficznego z F. Klasę grafów zdefiniowaną przez podanie zbioru $\mathcal{F} = \{F_1, \ldots, F_r\}$ podgrafów zabronionych oznaczymy przez $\mathrm{Forb}(F_1 \ldots, F_r)$. W badaniach nad klasami grafów P_k-wolnych Gyárfás i Lehel [14] wykazali, że algorytm FF koloruje $\chi(G)$ kolorami dowolny graf $G \in \mathrm{Forb}(P_4)$. W tej samej pracy, dla dowolnego $n = 1, 2, \ldots$ podano konstrukcję grafu dwudzielnego $G_n \in \mathrm{Forb}(P_6)$, dla którego $\chi(G) = 2$, ale dla dowolnego algorytmu A kolorowania on-line istnieje taka on-line prezentacja grafu G_n, która wymusza użycie co najmniej n kolorów. Zatem dla grafów z klasy $\mathrm{Forb}(P_6)$ nie istnieje efektywny algorytm A kolorowania on-line. Lukę dla klasy $\mathrm{Forb}(P_5)$ wypełnili Gyárfás i Lehel [16], dowodząc istnienie efektywnego dla niej algorytmu. Autorzy postawili hipotezę, że dla klasy grafów P_5-wolnych efektywny jest także algorytm FF i wykazali, $\chi_{FF}(G) \leq 3$ dla dowolnego $G \in \mathrm{Forb}(P_5, C_3)$ (podklasa $\mathrm{Forb}(P_5)$). Hipotezę tę rozstrzygnęli twierdząco Kierstead i in. w pracy [21], zawierającej ponadto szereg rozszerzeń wcześniejszych rezultatów dla klas grafów charakteryzowanych przez zbiory podgrafów zabronionych. Bardzo ciekawe

wyniki dla algorytmów kolorowania on-line można otrzymać analizując strukturę grafów k-krytycznych. Graf G nazywamy k-krytycznym dla algorytmu A, jeżeli $\chi_A(G) = k$, ale $\chi_A(G - v) < k$ dla każdego $v \in V(G)$. Obecność podgrafu k-krytycznego dla algorytmu A gwarantuje istnienie takiej prezentacji grafu G, która wymusi na algorytmie A użycie przynajmniej k kolorów. Wyniki badań struktry grafów k-krytycznych podane są w pracach [13, 5].

Przegląd wielu podanych tu oszacowań oraz innych wyników i otwartych problemów znaleźć można w artykule przeglądowym [24], którego autorami są Kierstead i Trotter.

5. Zastosowania algorytmów kolorowania grafów w trybie on-line

Główną motywacją w poszukiwaniu metod rozwiązywania problemów w trybie on-line jest ich bardzo duża przydatność praktyczna. Natura wielu rzeczywistych problemów stanowi, że pełna sekwencja danych wejściowych (żądań) nie może być znana, a rozwiązanie musi przebiegać w trybie on-line w warunkach całkowitego braku informacji o przyszłych żądaniach. Wśród wielu zastosowań algorytmów on-line, znajdujemy zastosowania dotyczące systemów operacyjnych, systemów przetwarzania równoległego i rozproszonego, kompilatorów, robotyki, projektowania układów VLSI, przydziału rejestrów procesora, dynamicznego przydziału pamięci, szeregowania zadań i wiele innych.

Jednym z najczęściej cytowanych przykładów zastosowania kolorowania on-line jest *problem dynamicznego przydziału pamięci* - DSA (ang. dynamic storage allocation) sformułowany w [12] następująco:
Dany jest zbiór $A = \{a_1, a_2, \ldots, a_t\}$ elementów oraz liczba naturalna D, będąca rozmiarem przestrzeni do przechowywania elementów. Każdy element opisany jest trójką atrybutów, rozmiarem $s(a)$, czasem $r(a)$ rozpoczęcia i czasem $d(a)$ zakończenia przechowywania elementu. Czy istnieje funkcja $\sigma : A \to \{1, 2, \ldots, D\}$ taka, że przyporządkowany każdemu elementowi $a \in A$ przedział (przestrzeń do przechowania) $l(a) = [\sigma(a), \sigma(a)+1, \ldots, \sigma(a)+s(a) - 1]$ zawarty jest w $[1, D]$ i taka, że dla dowolnych $a, a' \in A$, jeżeli $l(a) \cap l(a') \neq \emptyset$, to albo $d(a) \leq r(a')$, albo $d(a') \leq r(a)$? Element a można interpretować jako zmienną w programie komputerowym, a przedział $[r(a), d(a)]$ jako czas przebywania zmiennej w pamięci. Problem dotyczy zatem istnienia przyporządkowania wszystkim zmiennym $a \in A$ bloków pamięci o rozmiarze $s(a)$ każdy, w dostępnej pamięci o całkowitym rozmiarze D tak, że bloki przyporządkowane dowolnym dwóm zmiennym przebywającym w pamięci

w tym samym czasie nie nakładają się. W ogólnym przypadku, jak wykazał Stockmeyer, problem DSA jest silnie NP-zupełny. Niektóre dolne ograniczenia związane z tym problemem opisane są w pracy [6]. Szczególny przypadek problemu DSA występuje, gdy $s(a_i) = s(a_j)$. Jest on równoważny problemowi kolorowania grafu przedziałów i może być rozwiązany w trybie off-line w czasie wielomianowym przez algorytm First-Fit, gdy przedziały uporządkowane są względem ich lewych końców.

Kolejnym przykładem zastosowania kolorowania on-line jest podany w [11] następujący *problem szeregowania zadań w trybie on-line* (ang. on-line scheduling problem):
Danych jest k identycznych, równoległych procesorów oraz zbiór niepodzielnych zadań $Z = \{z_1, z_2, \ldots\}$. W momencie t_i przybycia zadania z_i do systemu zgłaszany jest czas $p_i \geq 0$ potrzebny do jego obsługi. W celu rozpoczęcia obsługi nowego zadania, obsługa dowolnego z aktualnie wykonywanych zadań może zostać przerwana bez ponoszenia dodatkowych kosztów. Zadanie, które nie zostanie natychmiast przypisane do procesora oraz zadanie, którego obsługa zostanie przerwana, nazywane jest *zadaniem straconym*. Problem polega na wyznaczeniu takiego przyporządkowania procesorom zadań, które minimalizuje liczbę zadań straconych.

Zdefiniujmy graf $G = (Z, E)$, gdzie $z_iz_j \in E, i \neq j$, gdy dla przedziałów czasowych $I_{z_i} = [t_i, t_i + p_i]$, $I_{z_j} = [t_j, t_j + p_j]$ mamy $I_{z_i} \cap I_{z_j} \neq \emptyset$. Graf G jest grafem przedziałów rodziny $\mathcal{I} = \{I_{z_i} : z_i \in Z(G)\}$. Wówczas minimalizacja liczby straconych zadań równoważna jest wyznaczeniu największego, w sensie liczby wierzchołków, podgrafu, dla którego istnieje k-pokolorowanie. Faigle i Nawijn [11] podają prosty, optymalny algorytm on-line rozwiązujący powyższy problem.

6. Inne metody oceny efektywności algorytmów kolorowania on-line

W opinii wielu badaczy zastosowanie metod probabilistycznych w ocenie algorytmów prowadzi do oszacowań lepiej oddających rzeczywiste zachowanie się algorytmu w zastosowaniach niż pesymistyczna analiza najgorszego przypadku. Kluczem do tego rozumowania jest niejednokrotnie uzasadniona nadzieja, co potwierdzają również badania eksperymentalne uzasadniające, że prawdopodobieństwo pojawienia się na wejściu algorytmu najgorszych konfiguracji danych jest niewielkie. Zastosowanie analizy najgorszego przypadku może prowadzić do podobnych oszacowań teoretycznych dla dwóch algorytmów, które w praktyce dają rozwiązania o bardzo różnej jakości. Przyczyną tak dużej rozbieżności

pomiędzy teorią i praktyką jest, jak piszą Irani i Karlin [18], przyjęty model. Trudno nazwać wyrównanym współzawodnictwo pomiędzy algorytmem on-line, nie posiadającym żadnych informacji o przyszłych żądaniach, a prezenterem posiadającym nieograniczone zasoby i informacje. Autorki analizują także stosowane metody ograniczania możliwości prezentera lub dostarczania algorytmowi dodatkowych informacji.

Jeden z najnowszych wyników znajdujemy w pracy [17], w której Halldórsson podaje algorytm probabilistyczny (ang. randomized) o funkcji dobroci $\rho_A(n) = O(n/\log n)$, poprawiając rezultat $O(n/\sqrt{\log n})$ autorstwa Vishwanathana [32]. Dla algorytmu First-Fit wymienimy oszacowanie $\chi_{FF}(G) \leq (2+\epsilon)\chi(G), \epsilon \geq 0$, które podał McDiarmid w [30], zachodzące dla prawie wszystkich grafów G. Metody probabilistyczne stosowali także Kučera [27] dla dolnego oszacowania dla algorytmu *FF*, Anthony i Biggs [1] dla oszacowania oczekiwanej liczby kolorów użytej przez *FF*, gdy wystąpienie każdej z on-line prezentacji grafu G jest jednakowo prawdopodobne. Jak podają Gyárfás i Lehel [14], interesujące wnioski wypływają także z prac eksperymentalnych Liska potwierdzających praktyczną przydatność algorytmu *FF* do kolorowania grafów przedziałów (r_{FF} pomiędzy 1.3 a 1.4).

Jednak nie wszędzie można stosować modele probabilistyczne. W sytuacjach gdy wymagana jest gwarancja jakości generowanych przez algorytm rozwiązań, stosowane są oszacowania pesymistyczne.

LITERATURA

1. M. Anthony, N. Biggs, *The mean chromatic number of paths and cycles*, Discrete Math. **120** (1993) 227–231.
2. D.R. Bean, *Effective coloration*, J. Symbolic Logic 41 (1976) 469–480.
3. L.L. Beck, D.W. Matula, *Smallest-Last Ordering and Clustering and Graph Coloring Algorithms*, J. ACM 30, 1 (Jul 1983) 417–427.
4. N. Biggs, *Some heuristics for graph colouring*, in: R. Nelson, R.J. Wilson eds., Graph colorings, Longman, New York, 1990, 87–96.
5. P. Borowiecki, *Characterization of Graphs Critical for First-Fit Graph Coloring*, in: 13 Workshop on Discrete Optimization, 27-30 April 1998, Burg (to appear).
6. D.J. Brown, B.S. Baker, H.P. Katseff, *Lower Bounds for On-Line Two-Dimensional Packing Algorithms*, Acta Informatica 18 (1982) 207–225.
7. M. Chrobak, M. Ślusarek, *Problem 84-23*, J. Algorithms 5 (1984) p.588.
8. M. Chrobak, M. Ślusarek, *On some packing problem related to Dynamic Storage Allocation*, RAIRO Informatique Théorique et Applications, 22 (1988) 487–499.
9. R. Diestel, Graph Theory, Springer-Verlag, 1997.

10. F.D.J. Dunstan, *Sequential colourings of graphs*, in: Proc. Fifth British Combinatorial Conf., *Utilitas Mathematica*, Winnipeg, 1976, 151–158.
11. U. Faigle, W.M. Nawijn, *Note on scheduling intervals on-line*, Discrete Appl. Math. **58** (1995) 13–17.
12. M.R. Garey, D.S. Johnson, Computers and Intractability, A Guide to the Theory of NP-Completeness, W.H. Freeman, San Francisco, 1979.
13. A. Gyárfás, Z. Király, J. Lehel, *On-line 3-chromatic graphs - II. Critical graphs*, Discrete Math. **177** (1997) 99–122.
14. A. Gyárfás, J. Lehel, *On-line and First Fit coloring of graphs*, J. Graph Theory **12** (1988) 217–227.
15. A. Gyárfás, J. Lehel, *On the special case of the wall problem*, Congr. Numerantium **67** (1988) 167–174.
16. A. Gyárfás, J. Lehel, *Effective on-line coloring of P_5-free graphs*, Combinatorica **11** (2) (1991) 181–184.
17. M.M. Halldórsson, *Parallel and On-line Graph Coloring*, J. Algorithms **23** (1997) 265–280.
18. S. Irani, A.R. Karlin, *Online Computation*, in: D.S. Hochbaum ed., Approximation algorithms for NP-hard problems, PWS Publishing Company, 1997, 521–564.
19. T.R. Jensen, B. Toft, Graph coloring problems, Wiley-Interscience series in discrete mathematics and optimization, Wiley, 1995.
20. H.A. Kierstead, *The linearity of First-Fit coloring of interval graphs*, SIAM J. Disc. Math. **1** (1988) 526–530.
21. H.A. Kierstead, S.G. Penrice, W.T. Trotter, *On-line and First-Fit coloring of graphs that do not induce P_5*, SIAM J. Disc. Math. **8** (1995) 485–498.
22. H.A. Kierstead, Jun Qin, *Coloring interval graphs with First-Fit*, Discrete Math., **144** (1995) 47–57.
23. H.A. Kierstead, W.T. Trotter, *An extremal problem in recursive combinatorics*, Congressus Numerantium **33** (1981) 143–153.
24. H.A. Kierstead, W.T. Trotter, *On-line graph coloring*, in: L.A. McGeoch and D.D. Sleator eds., On-line Algorithms, DIMACS Series in Discrete Math. and Comp. Sci. **7** ACM (1992) 85–92.
25. M. Kubale, *Problem kolorowania wierzchołków grafów. Przegląd algorytmów i zastosowań*, Zeszyty Naukowe Politechniki Śląskiej, Automatyka, z.144 (1994) 187–198.
26. M. Kubale, *Problem kolorowania krawędzi grafów. Przegląd algorytmów i zastosowań*, Zeszyty Naukowe Politechniki Śląskiej, Automatyka, z.117 (1996) 203–212.
27. L. Kučera, *The greedy coloring is a bad probabilistic algorithm*, J. Algorithms **12** (1991) 674–684.
28. L. Lovász, M. Saks, W.T. Trotter, *An on-line graph coloring algorithm with sublinear performance ratio*, Discrete Math. **75** (1989) 319–325.
29. D.W. Matula, G. Marble and J.D. Isaacson, *Graph coloring algorithms*, in: R.C. Read ed., Graph Theory and Computing, Academic Press, 1972, 109–122.
30. C.J.H. McDiarmid, *Coloring random graphs badly*, in: R.J. Wilson ed., Graph Theory and Combinatorics, Pitman, 1979, 76–86.
31. M. Ślusarek, *A Lower Bound for the First-Fit Coloring of Interval Graphs*, Zeszyty Naukowe Uniwersytetu Jagiellońskiego, Prace Informatyczne z.5 (1993) 25–32.

32. S. Vishwanathan, *Randomized online graph coloring*, J. Algorithms **13** (1992) 657–669.
33. A. Wigderson, *Improving the Performance Guarantee for Approximate Graph Coloring*, J. ACM 30, 4 (Oct 1983) 729–735.
34. H.S. Witsenhausen, *On Woodall's interval problem*, J. Combin Theory Ser. A, 21 (1976) 222–229.
35. D.R. Woodall *Problem no. 4*, in: T.P. McDonough and V.C. Mavron eds., Combinatorics: Proc. British Combinatorial Conference, Cambridge University Press, 1973, p.202.

Recenzent: Prof.dr hab.inż. Marek Kubale

Abstract

An on-line algorithm is one that receives a sequence of requests and, performs an immediate action in response to each request. It is assumed that the entire sequence is not known in advance. It is an inherent character of many real problems, that they have to be solved on-line. Any off-line algorithm has a limited power to solve such problems in practice.

In this paper we investigate the problem of on-line graph coloring. The most important results and bounds on the on-line chromatic number are reviewed. An application examples in Dynamic Storage Allocation and scheduling of basic on-line algorithms for graph coloring are included.