ZESZYTY NAUKOWE WYDZIAŁU ETI POLITECHNIKI GDAŃSKIEJ
Nr 4 Seria: Technologie Informacyjne 2006

Adrian Kosowski, Łukasz Kuszner

Katedra Algorytmów i Modelowania Systemów, Politechnika Gdańska

ZACHŁANNE ALGORYTMY KOLOROWANIA GRAFÓW W MODELU ROZPROSZONYM

Streszczenie

W artykule rozważamy problem kolorowania grafów w modelu rozproszonym. Głównym jego założeniem jest, że każdy wierzchołek stanowi odrębną jednostkę obliczeniowa, która może komunikować się tylko ze swoimi sąsiadami w grafie. W porównaniu ze standardowym, sekwencyjnym modelem kolorowania mamy tu dwie podstawowe różnice: obliczenia wykonują się w każdym z wierzchołków, a każdy z nich zna wyłącznie swoich sąsiadów i nie posiada żadnych danych o globalnej strukturze całego grafu. Przedstawiamy nowe modyfikacje znanego wcześniej algorytmu DLF i wyniki eksperymentów obliczeniowych.

1. WSTĘP

Problemy identyfikacji lokalnych i globalnych własności problemów kombinatorycznych w modelu rozproszonym, a także konieczność użycia identyfikatorów albo randomizacji ze względu na występowanie symetrii cieszą się zainteresowaniem świata nauki od lat osiemdziesiątych ubiegłego stulecia [1, 2]. Są to zagadnienia fundamentalne dla teorii algorytmów rozproszonych w ogóle, a kolorowanie grafów, jako problem szeroko znany i łatwo zrozumiały, wydaje się dla nich dobrym źródłem przykładów. Z jednej strony warunek na legalność pokolorowania jest ściśle lokalny, wystarczy porównać kolory sąsiadujących wierzchołków, z drugiej strony kryterium optymalizacyjne jest globalne, gdyż należy użyć jak najmniej kolorów w całym grafie. Problem kolorowania rozproszonego jest atrakcyjny również ze względu na zastosowania w przyznawaniu kodów w sieciach bezprzewodowych [3, 4]. W tym wypadku wierzchołki grafu identyfikujemy ze stacjami nadawczo odbiorczymi, występowanie krawędzi odzwierciedla możliwość występowania zakłóceń pomiędzy odpowiednimi stacjami.

Głównym założeniem rozpatrywanego tu modelu kolorowania jest, że każdy wierzchołek stanowi odrębną jednostkę obliczeniowa, która może komunikować się tylko ze swoimi sąsiadami w grafie. W porównaniu ze standardowym modelem kolorowania mamy tu dwie podstawowe różnice: obliczenia wykonują się równolegle, a wiedza o grafie jest rozproszona w wierzchołkach – każdy z nich zna wyłącznie swoich sąsiadów i nie posiada żadnych danych o globalnej strukturze całego grafu.

2. DEFINICJA PROBLEMU I MODEL OBLICZEŃ

Niech $G=(V, E)$ będzie grafem prostym, w którym V jest zbiorem wierzchołków, a E zbiorem krawędzi. Poprzez *pokolorowanie* grafu G rozumiemy funkcję $c : V \rightarrow \mathrm{N}$, która przyporządkowuje wierzchołkom grafu liczby naturalne – kolory. Warunkiem *legalności* pokolorowania jest, aby dla wszystkich wierzchołków $(u, v) \in E$ spełniony był warunek $c(u) \neq \mathrm{c}(v)$, a więc sąsiednie wierzchołki muszą otrzymać różne kolory. Pokolorowanie, w którym zbiór c(V) zawiera k elementów nazywamy k-pokolorowaniem i żądamy, aby liczba użytych kolorów była jak możliwie mała. Najmniejsze takie k, że istniej k-pokolorowanie G nazywamy *liczbą chromatyczną* grafu G i zapisujemy $\chi(G)$. Oznaczmy teraz liczbę wierzchołków przez $n = |V|$, liczbę krawędzi przez $m = |E|$ oraz *otwarte sąsiedztwo* wierzchołka v przez $N(v) = \{u : (u, v) \in E\}$. *Stopniem* wierzchołka v nazywamy liczbę jego sąsiadów i oznaczamy $\deg(v) = |N(v)|$, natomiast przez Δ oznaczymy największy stopień wierzchołka w grafie. Zdefiniujemy też podzbiór sąsiedztwa otwartego v składający się z wierzchołków o nie mniejszym stopniu niż v i oznaczymy $N_{\geq}(v) = \{u : u \in N(v) \wedge \deg(u) \geq \deg(v)\}$.

Przyjmujemy stosowany wcześniej synchroniczny model [5, 6, 7], w którym obliczenia przebiegają w *rundach*. Każda runda składa się z fazy obliczeń lokalnych, po której następuje faza komunikacji. Dodatkowo przyjmuje się następujące uproszczenia: za miarę złożoności obliczeniowej służy wyłącznie liczba rund; zarówno obliczenia jak i komunikacja są niezawodne. *Rozproszoną implementacją sekwencyjnego* algorytmu kolorowania A grafu G będziemy nazywać taki rozproszony algorytm DA, dla którego każde otrzymane pokolorowanie grafu G może być również otrzymane przy użyciu algorytmu A.

3. DOTYCHCZASOWE WYNIKI BADAŃ

Na dotychczasowe badania nad kolorowaniem rozproszonym składa się kilka nurtów. Pierwszym jest chromatyczna teoria grafów zwłaszcza w ujęciu algorytmicznym; teoria przetwarzania równoległego, w tym równoległego kolorowania grafów i teoria przetwarzania rozproszonego. Pokrewnymi tematami są modele obliczeń w warunkach niepewności, jak na przykład model kolorowania on-line.

3.1 Kolorowanie sekwencyjne

Próby skonstruowania szybkiego algorytmu dokładnego zakończyły się niepowodzeniem. Karp w 1972 roku wykazał NP-zupełność decyzyjnej wersji problemu kolorowania [8]. Również poszukiwania dobrego algorytmu przybliżonego nie dały rezultatu, co więcej Bellare i inni w roku 1995 pokazali, że dla $\varepsilon < 1/7$ nie może nawet istnieć algorytm n^{ε}-względnie przybliżony, o ile $\mathrm{P} \neq \mathrm{NP}$ [9]. Stąd, konstrukcja algorytmu polilogarytmicznie przybliżonego jest NP-trudna. Po tych negatywnych wynikach bardziej zainteresowano się konstrukcją i analizą algorytmów o dobrej efektywności oczekiwanej oraz działających efektywnie z dużym prawdopodobieństwem. Starano się również rozwiązać problem dla węższych klas grafów. Między innymi analizowano *algorytmy zachłanne*. W przypadku kolorowania grafów istnieje cała rodzina sekwencyjnych algorytmów zachłannych. Algorytmy te składają się z dwóch faz:

1. Porządkowanie wierzchołków grafu;
2. Kolorowanie zgodnie z ustalonym porządkiem kolejno wszystkich wierzchołków używając każdorazowo najmniejszego dostępnego koloru.

Uporządkowanie wierzchołków możliwe jest na różne sposoby. W algorytmie naiwnym wierzchołki brane są w zupełnie przypadkowej kolejności, w algorytmie LF (ang. largest first) bierzemy wierzchołki w porządku nierosnących stopni. W algorytmie SL (ang. smallest last) ustawiamy wierzchołki $v_1, v_2,\ldots,v_n$ w ten sposób, że wierzchołek v_i ma najmniejszy stopień w grafie indukowanym przez $v_1, v_2,\ldots,v_i$. W algorytmie SLF (ang. saturation largest-first), lub inaczej DSATUR, kolejność jest ustalana już w czasie kolorowania i zależy od dotychczasowego przydziału kolorów. Podstawowym kryterium jest tu liczba kolorów użytych przez sąsiadów danego wierzchołka. Wszystkie te metody mają liniową funkcję dobroci, tj. ich stosunek liczby użytych kolorów do $\chi(G)$ może rosnąć proporcjonalnie do n.

3.2 Kolorowanie rozproszone

Prawdopodobnie pierwszy rozproszony polilogarytmiczny algorytm kolorowania grafów ogólnych, działający w czasie $O(\log^* n)$ podał Linial [6]. Jest to praktycznie rzecz biorąc czas stały, jednak algorytm ten używa aż $O(\Delta^2)$ kolorów. Ponadto zakłada się, że globalne parametry grafu: Δ i n są znane w każdym z wierzchołków.
Johansson w pracy [10] w elegancki sposób wykazał, że modyfikacja algorytmu Luby'ego [11] działa prawie zawsze w czasie $O(\log n)$. Luby podał algorytm, w którym każdy wierzchołek, na początku każdej rundy „budzi się" z prawdopodobieństwem p=1/2, następnie każdy obudzony wierzchołek losuje sobie barwę z palety dostępnych kolorów. Jeśli wylosowany kolor nie koliduje z kolorami wybranymi przez sąsiadów, to wierzchołek kończy działanie, w przeciwnym wypadku usuwa z palety kolory wybrane przez sąsiadów i podejmuje kolejną próbę w następnej rundzie. Johansson wyeliminował losową fazę „budzenia się", pozostawiając losowy wybór koloru w fazie kolejnej. Oba algorytmy używają co najwyżej Δ+1 kolorów. Eksperymenty komputerowe pokazują, że modyfikacja wprowadzona przez Johanssona przyspiesza proces bez utraty jakości pokolorowania [12]. W kolejnej pracy Panconesi i Rizzi [13] podali algorytm oparty na ciekawej technice dekompozycji grafu na las ukorzenionych drzew. Zakłada się tam istnienie identyfikatorów wierzchołków, ale można również użyć wielkości losowych. Algorytm ten działa w czasie $O(\Delta^2 + \log^* n)$ i używa maksymalnie Δ+1 kolorów. Z kolei algorytmy DLF i DDLF oparte na idei zaczerpniętej z sekwencyjnego algorytmu LF opisano w [14, 15]. Działają one w czasie $O(\Delta^2 + \log n)$ i gwarantują pokolorowanie Δ+1 barwami. Wydaje się jednak, że analizę tych algorytmów można poprawić uzyskując lepsze oszacowanie zarówno ze względu na duże uproszczenia przyjęte w analizie jak i ze względu na wyniki przeprowadzonych eksperymentów.

4. ROZPROSZONE ALGORYTMY TYPU NAJWIĘKSZY NAJPIERW

Pokażemy teraz jak zmodyfikować algorytm DLF, aby otrzymać rozproszoną implementację LF. Zacznijmy od opisu samego algorytmu DLF. Każdy wierzchołek przechowuje następujące zmienne lokalne:

c – aktualny kolor wierzchołka,
d – *stopień* wierzchołka,
f – *zmienna* logiczna, ustawiona oznacza, koniec procesu w danym wierzchołku,
r – *wartość* losowa,
Nf – lista niepokolorowanych sąsiadów.

Algorytm 1: DLF

Runda 0: $f(v) := 0$;
$d(v) := \deg(v)$;
$r(v) := 0$;

Runda $2k$-1: **if** $f(v)$ = false **then**

k=1,2,… $Nf(v) := \{u : u \in N(v) \wedge f(u) = \text{false}\}$;
$c(v) := \min\{a \in \mathbb{N} : \forall(u \in N(v) \Rightarrow c(u) \neq a)\}$;
$r(v) := \text{random}()$;

Runda 2k **if** $f(v)$ = false **then**

k=1,2,… **if** $\forall\big((u \in Nf(v) \wedge c(u) = c(v)) \Rightarrow$
$(d(u)<d(v) \vee (d(u) = d(v) \wedge r(u) < r(v)))\big)$ **then** $f(v) := \text{true}$;

Runda o numerze 0 ma charakter przygotowawczy. W rundach nieparzystych zapamiętywane są wierzchołki, które nie zakończyły jeszcze kolorowania; zmienna c zostaje ustawiona na pierwszy kolor nie zajęty jeszcze przez żaden wierzchołek, który uzyskał już kolor; zmienna r przyjmuje wartość przypadkową (w testach posłużyliśmy się 32 bitowymi liczbami całkowitymi). W rundach parzystych wierzchołki, które spośród sąsiadów próbujących uzyskać ten sam kolor (identyczna wartość zmiennej c mają największy stopień, lub maksymalny stopień i największą wartość r) zachowują kolor i kończą działanie – zmienna f równa true. Jak podano w pracy [14] algorytm ten nie jest rozproszoną implementacją algorytmu LF, gdyż możliwa jest następująca sytuacja: $c(v) < c(u)$, u i v są sąsiednie, $\deg(v) < \deg(u)$ oraz $c(v) \notin \{c(x) : x \in N_{\geq}(u)\}$. Można tego uniknąć wprowadzając w rundach parzystych dodatkowy warunek na zakończenie procesu kolorowania, a mianowicie wierzchołek nie może uzyskać koloru, jeśli ma sąsiada o wyższym stopniu, który próbuje uzyskać mniejszy kolor. Bardziej formalnie zapiszemy to w postaci poniższego algorytmu:

Algorytm 2: DLFa

Runda 0: $f(v) := 0$;
$d(v) := \deg(v)$;
$r(v) := 0$;

Runda $2k$-1: **if** $f(v)$ = false **then**

k=1,2,… $Nf(v) := \{u : u \in N(v) \wedge f(u) = \text{false}\}$;
$c(v) := \min\{a \in \mathbb{N} : \forall(u \in N(v) \Rightarrow c(u) \neq a)\}$;
$r(v) := \text{random}()$;

Runda 2k **if 2,·f**(v) = false **then**

k=1,2,… **if** $\forall\big(u \in Nf(v) \Rightarrow (d(u) < d(v) \vee$
$(d(u) = d(v) \wedge r(u) < r(v)) \vee$
$(d(u) > d(v) \wedge (c(u)>c(v)))\big)$ **then** $f(v) := \text{true}$;

Kolejną modyfikacją obu algorytmów może być uwzględnienie zmieniającej się struktury grafu i branie w charakterze stopnia wierzchołka liczby niepokolorowanych jeszcze sąsiadów. W tym celu wystarczy w rundach nieparzystych podstawić pod d liczbę wierzchołków w Nf. Otrzymamy w ten sposób dwa kolejne warianty algorytmu DLF. Będziemy je nazywać odpowiednio DDFL i DDLFa.

5. WYNIKI EKSPERYMENTÓW

W celu porównania przedstawionych algorytmów dokonaliśmy wielokrotnych symulacji ich wykonań. Uzyskane wyniki przedstawiono w poniższych tabelach. W każdym wierszy umieszczono uśrednione wyniki testów wykonywanych po 10 na każdym z 10 losowych grafów o zdanych parametrach. Parametrami grafów są n – liczba wierzchołków oraz $\varphi = 2m/n$ – średni stopień wierzchołka lub $\rho = 2m/n(n-1)$ – gęstość grafu. W odpowiednich kolumnach umieszczono: Δ – średni największy stopień wierzchołka, C_A – średnią liczbę kolorów użytych przez algorytm A, T_A – średnią liczbę rund użytych przez algorytm A.

Tablica 5.1

Liczba kolorów w grafach o stałej gęstości krawędzi

n	φ	Δ	C_{LF}	C_{DLF}	C_{DDLF}	C_{DLFa}	C_{DDLFa}
10	0,1	2,3	2,0	2,0	2,0	2,0	2,0
50	0,1	9,8	4,5	4,5	4,6	4,5	4,5
250	0,1	38,7	11,3	11,4	10,8	11,5	11,5
1250	0,1	160,7	34,8	34,9	32,9	34,9	34,7
10	0,5	6,6	3,9	3,9	3,9	3,9	3,9
50	0,5	32,0	12,1	12,1	11,6	12,0	12,1
250	0,5	146,7	40,0	40,2	38,3	40,3	40,0
1250	0,5	683,7	147,5	147,6	142,6	147,4	147,8
10	0,9	9,0	7,0	7,0	7,0	7,0	7,0
50	0,9	48,1	24,5	24,4	23,8	24,5	24,4
250	0,9	236,0	94,5	95,2	92,0	94,9	94,6
1250	0,9	1156.8	380,5	381,2	371,8	380,6	380,8

Tablica 5.2

Liczba kolorów w grafach o stałym średnim stopniu wierzchołka

n	ρ	Δ	C_{LF}	C_{DLF}	C_{DDLF}	C_{DLFa}	C_{DDLFa}
200	10	19,9	6,5	6,8	6,6	6,6	6,7
1000	10	22,2	7,0	7,0	7,0	7,0	7,0
5000	10	23,7	7,0	7,1	7,0	7,1	7,1
25000	10	25,6	7,2	7,6	7,0	7,2	7,3
200	20	32,5	9,8	9,9	9,3	9,8	9,9
1000	20	37,5	10,1	10,2	9,7	10,1	10,1
5000	20	38,3	10,3	10,5	10,0	10,4	10,3
25000	20	40,8	10,8	10,9	10,0	10,8	10,8
200	40	56,5	15,6	15,7	14,7	15,6	15,5
1000	40	61,3	15,4	15,6	14,6	15,5	15,4
5000	40	65,6	15,9	16,0	15,0	15,8	15,8
25000	40	68,9	16,1	16,1	15,0	16,0	16,1

Tablica 5.3

Liczba rund w grafach o stałej gęstości krawędzi

n	φ	Δ	T_{DLF}	T_{DDLF}	T_{DLFa}	T_{DDLFa}
10	0,1	2,3	5,1	5,1	5,1	5,1
50	0,1	9,8	10,4	10,6	10,9	10,9
250	0,1	38,7	24,0	23,0	27,0	27,0
1250	0,1	160,7	71,0	67,2	79,0	78,2
10	0,5	6,6	8,9	8,9	8,9	9,0
50	0,5	32,0	25,5	24,3	25,7	25,8
250	0,5	146,7	81,4	77,7	83,4	82,9
1250	0,5	683,7	296,3	286,5	299,6	299,7
10	0,9	9,0	15,0	15,0	15,1	15,1
50	0,9	48,1	49,9	48,6	50,1	49,9
250	0,9	236,0	191,5	185,1	190,9	190,3
1250	0,9	1156.8	763.4	744.6	762.3	762.8

Tablica 5.4

Liczba rund w grafach o stałym średnim stopniu wierzchołka

n	ρ	Δ	T_{DLF}	T_{DDLF}	T_{DLFa}	T_{DDLFa}
200	10	19,9	14,8	14,6	16,3	16,3
1000	10	22,2	15,2	15,0	17,9	17,9
5000	10	23,7	15,7	15,1	19,0	19,0
25000	10	25,6	16,9	15,6	19,5	19,3
200	20	32,5	20,9	19,9	23,0	23,1
1000	20	37,5	21,5	20,8	25,3	25,4
5000	20	38,3	22,4	21,0	26,8	26,7
25000	20	40,8	23,0	21,4	27,6	27,5
200	40	56,5	32,4	30,7	35,4	35,4
1000	40	61,3	32,4	30,6	37,6	37,5
5000	40	65,6	33,1	31,1	39,5	39,4
25000	40	68,9	33,3	31,1	40,9	40,8

6. ZAKOŃCZENIE

Przedstawione wyniki pokazują, że zaproponowana strategia czekania na wierzchołki o większym stopniu i mniejszym kolorze nieznacznie poprawia liczbę użytych kolorów kosztem również nieznacznego zwiększenia liczby rund. Ciekawym wydaje się fakt, iż wzięcie w charakterze stopni wierzchołka liczby niepokolorowanych sąsiadów znacznie poprawia wyniki w przypadku algorytmu DDLF w stosunku do DLF, natomiast w przypadku DDLFa i DLFa obserwujemy znacznie mniejszą różnicę.

Wyniki potwierdzają również przypuszczenie, że uzyskane w pracy [12] ograniczenie na czas działania algorytmu DLF może być grube i być może da się je jeszcze poprawić.

BIBLIOGRAFIA

[1] Awerbuch B.: *A new distributed depth-first-search algorithm*, Information Processing Letters 20 (1985), 147–150.
[2] Johnson R. E., Schneider F. B.: *Symmetry and similarity in distributed systems*. W: Proceedings of PODC 1985, 13–22.
[3] Gandham S., Dawande M., Prakash R.: *Link scheduling in sensor networks: Distributed edge coloring revisited*. W: Proceedings of INFOCOM 2005 (w druku).
[4] Sousa E. S., Silvester J. A.: *Spreading code protocols for distributed spread-spectrum packet radio networks*, IEEE Transactions on Communications 36 (1988), 272–281.
[5] Chaudhuri, P.: *Algorithms for some graph problems on a distributed computational mode.* Information Sciences 43 (1987), 205–228.
[6] Linial, N.: *Locality in distributed graph algorithms*, SIAM J. Computing 21 (1992), 193–201.
[7] Panconesi, A., Srinivasan, A.: *Improved distributed algorithms for coloring and network decomposition problems*, W: Proceedings of STOC 1992, 581–592.
[8] Karp R. M.: *Complexity of computer computations*. W: Reducibility Among Combinatorial Problems, Plenum Press, 1972, 85–103.
[9] Bellare M., Goldreich O., Sudan M.: *Free bits, PCPs and nonapproximability – towards tight results*. W: Proceedings of FOCS 1995, 422.
[10] Johansson Ö.: *Simple distributed (Δ+1)–coloring of graphs*. Information Processing Letters 70, 1999, 229–232.
[11] Luby M.: *Removing randomness in parallel without processor penalty*. Journal of Computer and System Sciences 47, 1993, 250–286.
[12] Kuszner Ł.: Rozproszone kolorowanie grafów, WETI PG, 2001.
[13] Panconesi A., Rizzi R.: *Some simple distributed algorithms for sparse networks*. Distributed Computing 14, 2001, 97–100.
[14] Hansen J., Kubale M., Kuszner Ł., Nadolski A.: *Distributed largest-first algorithm for graph coloring.* W: Proceedings of Euro-Par 2004, 804–811.
[15] Janczewski R., Kuszner Ł., Małefijski M., A. Nadolski. An experimental study of distributed algorithms for graph coloring. W: Proceedings of ACS-SCM 2003, 295–298.
[16] Nowak K.: *Dydaktyczny model łączenia sieci LAN za pomocą sieci rozległych*, WETI PG, 2002.
[17] Kowalski A., Kabacki K.: *Simulation of Network Systems in Education*. W: Proceedings of the XXIV Autumn International Colloquium Advanced Simulation of Systems. ASIS 2002, September 9-11 2002, Ostrava, Czech Republic., s.213-218, bibliogr. 16 poz. Acta MOSIS nr 90.

GREEDY ALGORITHMS FOR DISTRIBUTED GRAPH COLOURING

Summary

In the paper we discuss graph colouring problem in a distributed model. The most important assumption is that each vertex is an independent processing unit and is able to communicate with its neighbours in the graph. In comparison to traditional, sequential graph colouring we have two important differences: computations are done independently in each vertex and the information about the structure of a graph is distributed among its vertices. Consequently each vertex knows only its neighbours and does not have access to any global parameters of a graph. We introduce some new modifications of previously known DLF algorithm and present the results of conducted experiments.