Wydział Podstawowych Problemów Techniki
Politechnika Wrocławska

Metody optymalizacji
Minimum Graph Coloring

Kamil Sikorski
Nr indeksu: 221481

Przedmiot prowadzony przez

Pawła Zielińskiego

Politechnika Wrocławska

Wrocław 2019

Spis treści

Wstęp

W *Teori Grafów* kolorowanie grafu jest szczególnym przypadkiem etykietowania, tradycyjnie nazywanymi kolorami. *Minimalne Kolorowanie Grafu* etykietuje wierzchołki (kolorami) tak bym, żadna krawędź nie łączyła dwóch wierzchołków o tej samej etykiecie (kolorze). Minimalna ilość kolorów potrzebna do *Minimalnego Kolorowania Grafu* nazywana jest *Liczbą Chromantyczną* samo wyznaczenie tej liczby jest problemem *NP-trudnym*[8], dlatego nie ma wielomianowego algorytmu rozwiązującego zadanie.

Zastosowaniem algorytmu rozwiązującego kolorowanie grafu, może być szeregowanie zadań. Gdzie wierzchołki są zadaniami, a krawędzie pomiędzy nimi zasobem współdzielonym. Rozwiązanie problemu mówiło by w jaki sposób uruchamiać grupy zadań, by zasób wspódzielony wykorzystywany byłby przez jedną maszynę na raz.

Problemem z życia może być układanie planu zajęć, gdzie krawędziami są sale lub lektorzy, a wierzchołkami ilość zajęć. Liczba kolorów mówiłaby ile godzin zajęciowych potrzebnych jest do zrealizowania planu, a grupy wierzchołków prezentowały by jakie zajęcia mogą obywać się o jednej godzinie.

1.1 Opis problemu

Minimalne Kolorowanie Grafu jest przypisaniem kolorów do każdego wierzchołka grafu, tak że żadna krawędź nie łączy dwóch identycznie pokolorowanych wierzchołków. Problemem jest znalezienie takiego etykietowania by nie można było znaleźć etykietowania z mniejszą ilością kolorów.

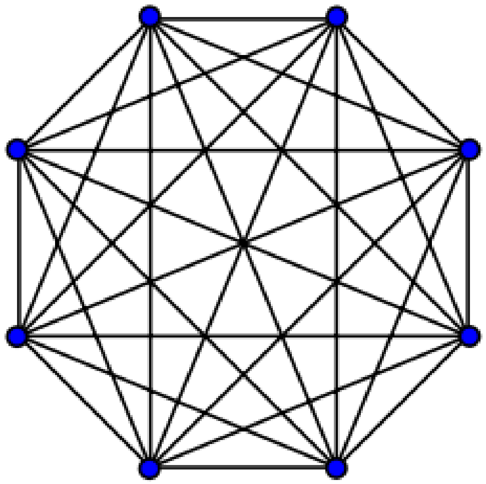
1.1.1 Model

- Dane: Graf $G(V, E)$ gdzie V to wierzchołki, a E to krawędzie.
- Rozwiązanie: Kolorowanie G, tj. podział wierzchołków V na zbiory $V_1, V_2, ..., V_k$, takie że każdy zbiór V_i jest niezależny w G.
- Miara: Ilość zbiorów V_i

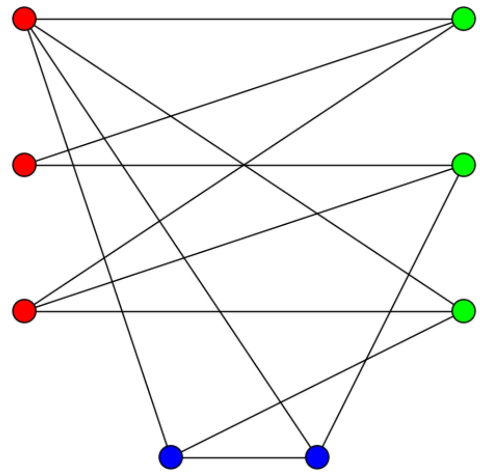
1.2 Przykłady

Poniżej zostały przedstawione przykłady rozwiązania problemu *Minimalnego Kolorowania Grafu* z różnymi rodzajami grafów. Omawiane grafy przedstawione są na rysunkach 1.1.

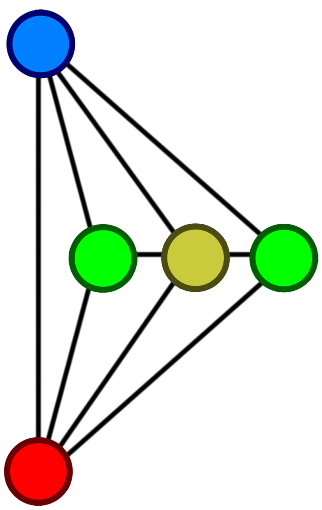
- *Graf pełny* rozwiązaniem dla niego jest $|V|$, ponieważ każdy wierzchołek posiada krawędź z pozostałymi wierzchołkami, dlatego każdy z nich musi mieć inny kolor.
- *Graf k-dzielny* jego definicja opisuje szukane rozwiązanie dla *Minimalnego Kolorowania Grafu*, tzn. wartość *k k-dzielności grafu* opisuje rozwiązanie. W rysunku *b* został przedstawiony graf *graf 3-dzielny*, gdzie kolory wierzchołków opisują grupy do której należą.
- *Graf planarny* każdy graf prosty tego rodzaju jest co najwyżej $4 - kolorowalny$ [1]. Na przedstawionym przykładzie grafu planarnego widać, że nie można znaleźć miejsca dla wierzchołka połączonego z czteroma wierzchołkami o różnych kolorach, nie przecinając istniejącej krawędzi.
- Dla pozostałych grafów przedstawione zostało górne ograniczenie *Kolorowanie grafu*, znane jako *Twierdzenie Brooksa* [3]. Ustala że *liczba chromatyczna* jest niewiększa niż maksymalny stopień wierzchołka $(max(deg(V)))$, z wyjątkiem *grafu z nieparzystym cyklem* wtedy wartość jest nie większa niż $max(deg(V)) + 1$.



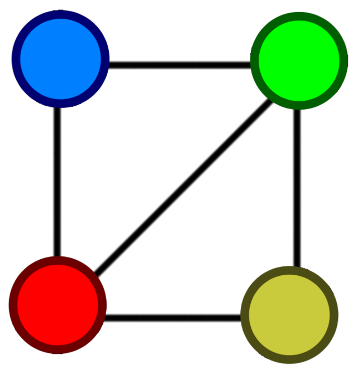
(a) Graf Pełny o 8 wierzchołkach - rozwiązaniem jest 8



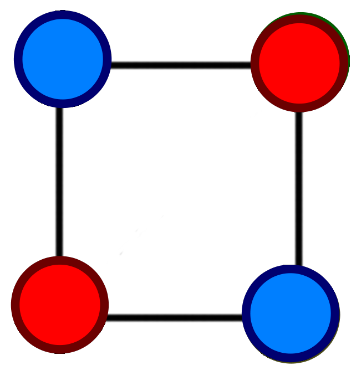
(b) Graf 3-dzielny - rozwiązaniem jego jest 3



(c) Graf planarny - rozwiązaniem jego jest 4



(d) Graf z cyklem nieparzystym - maksymalny stopień 3, rozwiązaniem jego jest 4



(e) Graf bez cyklu nieparzystego - maksymalny stopień 2, rozwiązaniem jego jest 2

Rysunek 1.1: Przykładowe grafy

Sposoby rozwiązania

2.1 Programowanie dynamiczne

Poniżej przedstawione algorytmy rozwiązują zagadnienie *Minimalnego kolorowanie grafu*, metodą dynamiczną, które rozwiązują problem dokładnie. Pierwszym algorytmem jest rozwiązanie przedstawione przez Lawlera. Znajduje optymalne rozwiązanie w czasie $O(2.4423^n)$, gdzie n to ilość wierzchołków[9]. Lepszą złożonością obliczeniową wykazał się algorytm Eppstein'a rozwiązuje on problem w czasie $O(2.4150^n)$ [5]. Kolejnym krokiem wykazał się Byskov i osiągnął jak narazie najlepszy wynik dla *Minimalnego Kolorowania Grafu* $O(2.4023^n)$ [4].

2.2 Programowanie zachłanne

Omówione poniżej algorytmy nie zapewniają optymalnego rozwiązania, często nie są jego bliskie, za to czas wykonywania jest znacznie szybszy. Zaletą tego rodzaju algorytmów jest to, że mogą posłużyć do znalezienia górnego ograniczenia.

Algorytm *DSATUR* zaproponowany przez Brélaz'a [2]. Alogorytm ten jest wciąż rozwijany oraz wykorzystywany, dlatego kod podstawowej wersji został umieszczony Algorithm 2.1, złożoność obliczeniowa wynosi $O(n^2)$. Innym algorytmem zachłannym jest *Recursive Largest First* (*RLF*), zaprojektowanym przez *Leighton* [10]. Algorytm działa w czasie $O(n^3)$, jest wolniejszy niż *DSATUR*, a wyniki są porównywalne.

2.3 DSATUR-based branch and bound

Metoda ta opiera się na przeszukiwaniu drzewa reprezentującego przestrzeń rozwiązań problemu. Stosowane w tej metodzie odcięcia redukują liczbę przeszukiwanych węzłów. *DSATUR-based branch-and-bound* w literaturze najczęściej spotyka się skróconą wewrsje *DSATUR*. W rozwiązaniu zasugewrował, aby rozpocząć kolorowanie od dużej kliki i etapu wstępnego przetwarzania przy użyciu heurystyki *DSATUR*. *Sewell* poprawił to podejście, wprowadzając nową strategię rozsądzania remisu [12]. *Segundo* zasugerował zastosowanie tej strategii tylko selektywnie, dzięki temu poprawił ogólną wydajność. Nazwał ten algorytm *PASS* i został testowany na podzbiorze *DIMACS*(informacje) i na grafach losowych. Praca *Fabio Furini, Virginie Gabrel, Ian-Christopher Ternier* polegała na zmodyfikowania podstawowej wersji by zmieniać dolną granice rozwiązania. Wyniki są lepsze dla grafów losowych o dużej gęstości $0.7 - 0.9$ [6], na grafach *DIMACS* uzyskuje podobne wyniki jak algorytm *PASS*.

Algorithm 2.1: DSATUR

Data: graf G
Result: Niezależne zbiory $\mathbf{S}$

$\mathbf{S} \rightarrow \emptyset$
$X \rightarrow V(G)$
while $X \neq \emptyset$ **do**
  Wybierz $v \in X$
  for $j \leftarrow 1 do |\mathbf{S}|$ **do**
    if $S_j \cup \{v\}$ *jest niezależny* **then**
      $S_j \leftarrow S_j \cup \{v\}$
      break
  if $j > |\mathbf{S}|$ **then**
    $S_j \leftarrow \{v\}$
    $\mathbf{S} \leftarrow \mathbf{S} \cup S_j$
  $X \leftarrow X - \{v\}$

2.4 Programowanie liniowe

Annuj Mehrotra, Michael A. Trick zaprezentowali sposób rozwiązania problemu programowaniem liniowym [11]. Metoda opiera się na rozwiązaniu problemu *Maximum weighted independent set* wielokrotnie i znalezieniu rozwiązania całkowitoliczbowego, które jest odpowiednie do uzyskania rozwiązania bazowego problemu.

Gualandi, Stefano and Malucelli, Federico rozwiązują problem łącząc wiele metod [7]. Podejście generowania kolumn zostało ulepszone dzięki zastosowaniu programowania ograniczeń w celu rozwiązania podprogramu cenowego i obliczenia rozwiązań heurystycznych. Ponadto wprowadzili nowe techniki w celu poprawy wydajności generowania kolumn w rozwiązywaniu zarówno liniowej relaksacji, jak i problemu liczby całkowitej. Dodatkowo rozszerzyli swoje rozwiązania do wyliczenia problemu *Minimum Vertex Graph Multicoloring*.

Bibliografia

[1] K. Appel, W. Haken. Every planar map is four colorable. *Bulletin of the American Mathematical Society*, strony 711–712, 1976.

[2] D. Brélaz. New methods to color the vertices of a graph. *Commun. ACM*, 22:251–256, 04 1979.

[3] R. L. Brooks. On colouring the nodes of a network. *Mathematical Proceedings of the Cambridge Philosophical Society*, strony 194–197, 1941.

[4] J. M. Byskov. Chromatic number in time o (2.4023ˆ n) using maximal independent sets. *BRICS Report Series*, 9(45), 2002.

[5] D. Eppstein. Small maximal independent sets and faster exact graph coloring. *J. Graph Algorithms & Applications*, 7(2):131–140, 2003. Special issue for WADS'01.

[6] F. Furini, V. Gabrel, I. Ternier. An Improved DSATUR-Based Branch-and-Bound Algorithm for the Vertex Coloring Problem. *Networks*, 69(1), 2017.

[7] S. Gualandi, F. Malucelli. Exact solution of graph coloring problems via constraint programming and column generation. *INFORMS Journal on Computing*, 24(1):81–100, 2012.

[8] R. M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, strony 85–103, 1972.

[9] E. Lawler. A note on the complexity of the chromatic number problem. *Information Processing Letters*, 5(3):66 – 67, 1976.

[10] R. R. Lewis. *A Guide to Graph Colouring: Algorithms and Applications*. Springer Publishing Company, Incorporated, wydanie 1st, 2015.

[11] A. Mehrotra, M. Trick. A column generation approach for graph coloring. *INFORMS J Comput*, 8, 09 1995.

[12] E. C. Sewell. A. improved algorithm for exact graph coloring. 26:359–373, 1993.