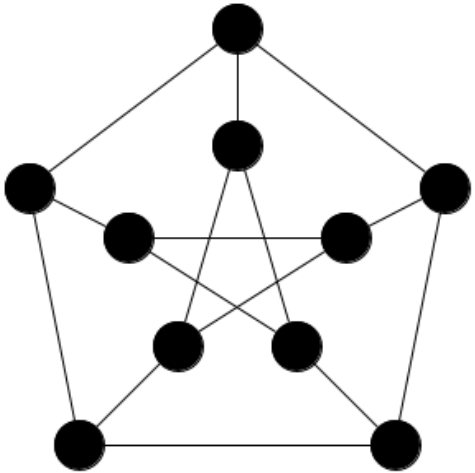
Вершинная раскраска графа методом МакКея

Абросимов Михаил Борисович
д.ф.-м.н., профессор, СГУ

Разумовский Петр Владимирович
аспирант, СГУ

Основные определения

Неориентированным графом (везде далее просто *графом*) называется пара $G = (V, \alpha)$, где α – симметричное и антирефлексивное отношение на множестве вершин V, называемое отношением *смежности*.

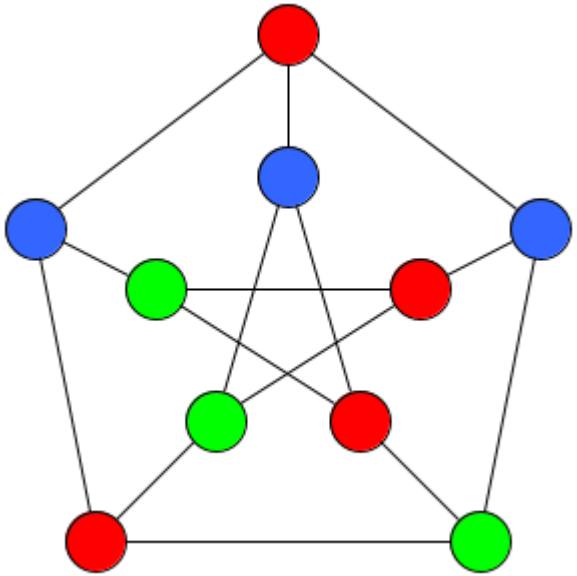


Основные определения

Пусть $G = (V, E)$ – граф, а $k \in \mathbb{N}$. Функция вида $f: V \to \{1, \dots, k\}$ называется *вершинной k-раскраской* графа G, а $f(v), v \in V$ – цветом вершины $v \in V$. При этом граф G называется *графом с цветными вершинами*, или *цветным графом.* Для таких графов вводится следующее обозначение: $G = (V, E, f)$.

Основные определения

Вершинная раскраска графа Петерсена:



Основные определения

Изоморфизмом цветных графов $G_1 = (V_1, E_1, f_1)$ и $G_2 = (V_2, E_2, f_2)$ называется изоморфизм графов $G_1 = (V_1, E_1)$ и $G_2 = (V_2, E_2)$, который сохраняет цвета. То есть это такая биекция $\varphi: V_1 \to V_2$, что выполняются два условия:

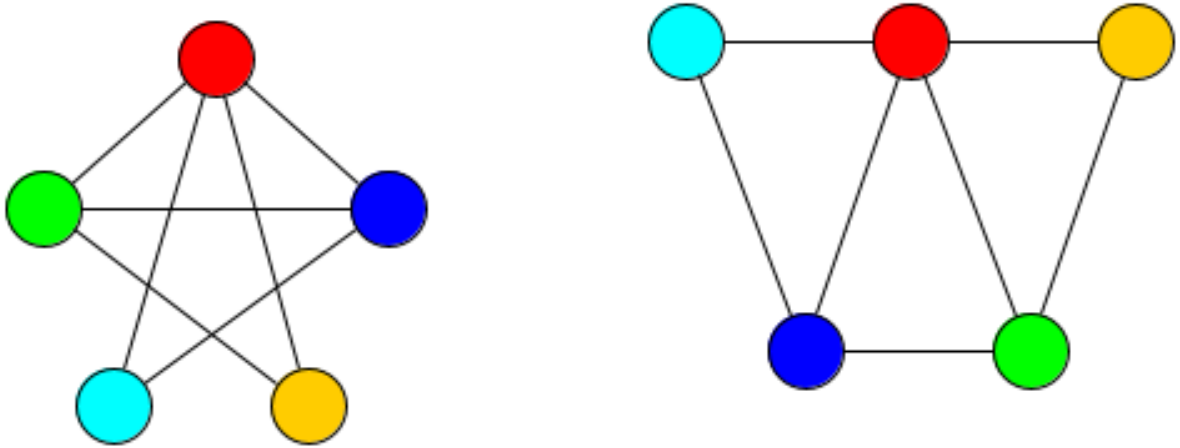
1) $(u, v) \in E_1 \Leftrightarrow \big(f(u), f(v)\big) \in E_2$ для любой пары вершин $u, v \in V_1$;

2) $f_1(v) = f_2\big(\varphi(v)\big)$ для любой вершины $v \in V_1$.

Автоморфизм (цветной) графа – изоморфизм (цветной) графа на себя.

Основные определения

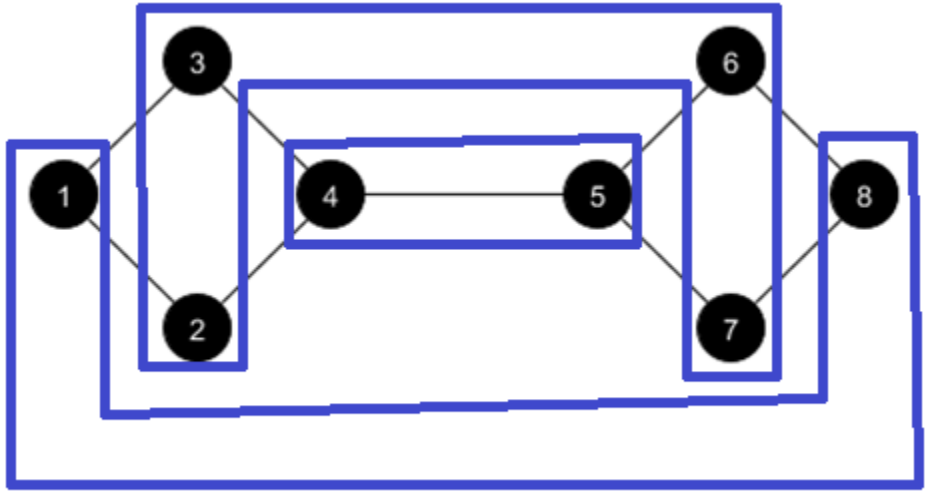
Пример двух изоморфных цветных графов:



Основные определения

Две вершины являются *подобными* тогда и только тогда, когда существует автоморфизм, который отображает одну вершину на другую ($\exists \varphi : u = \varphi(v)$).

Множество подобных вершин называется *орбитой.*



Данный граф имеет множество орбит:

$$\{\{1, 8\},\ \{2, 3, 6, 7\},\ \{4, 5\}\}$$

Алгоритм

Целью алгоритма является генерация всех вершинных k-раскрасок для заданного графа без проверки на изоморфизм.

Алгоритм

Алгоритм подразумевает конструирование канонических кодов – одинаковых для изоморфных цветных графов и разных для неизоморфных. Если на каком-то этапе конструирования очередного кода становится понятно, что код не будет каноническим – он отсекается. Также коды должны быть лексикографически минимальными.

Алгоритм

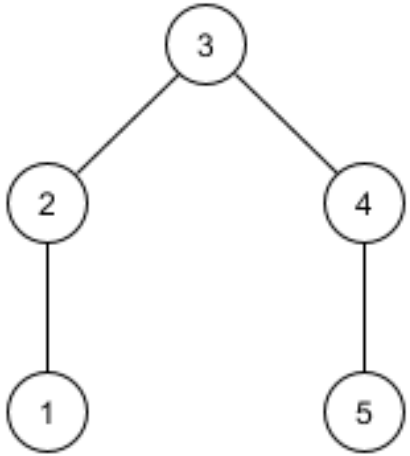
Общее описание метода МакКея может быть найдено в статье Graph canonical labeling and automorphism group computation // 2017. http://users.cecs.anu.edu.au/~bdm/nauty/nug26.pdf.

Множество орбит графа вычисляется при помощи программы *nauty,* ссылка на статью: McKay B. D., Piperno A. Practical Graph Isomorphism // J. Symbolic Computation. 2013. Vol. 2, 60. Pp. 94-112.

Алгоритм

Зададим граф $G = (V, E)$ и количество цветов k: $k \in [1; |V|]$.

Обозначим текущую раскраску вектором размерности n, инициализируем его элементы цветом 1 и вычислим орбиты для графа с данной раскраской.

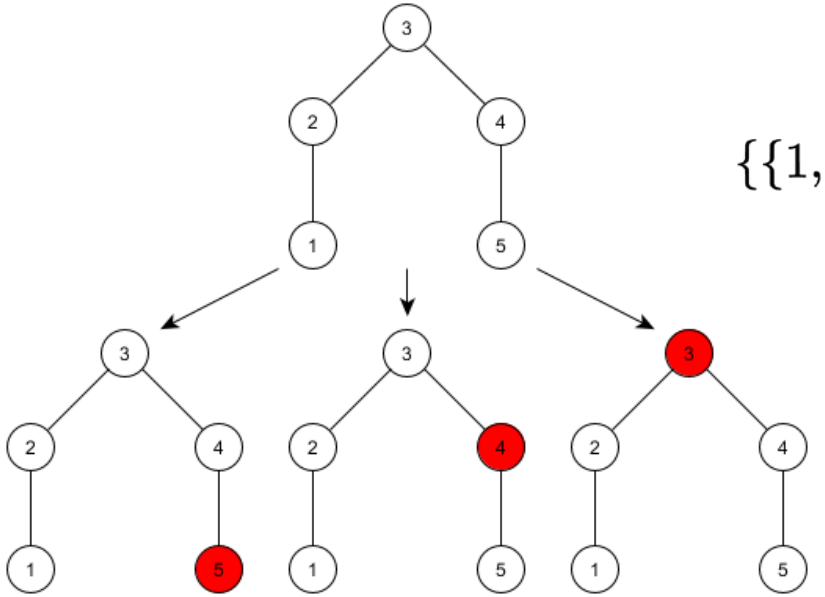


$$(1, 1, 1, 1, 1)$$

$$\{\{1, 5\}, \{2, 4\}, \{3\}\}$$

Алгоритм

Выберем вершины с наибольшим номером из каждой орбиты и последовательно раскрасим их цветами от 2 до k, получая новые раскраски.



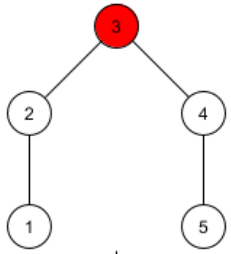
$$\{\{1,5\},\{2,4\},\{3\}\}$$

Алгоритм

Проверим полученные раскраски на каноничность. Если раскраска каноничная, то продолжаем генерацию, иначе отсекаем раскраску.

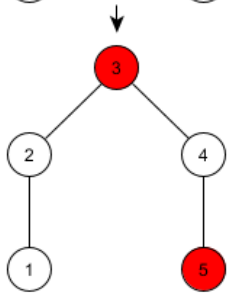
Имеем проверяемую раскраску и ее предка. Последовательно раскрашиваем каждую ее вершину в цвет 1 и сравниваем получившуюся раскраску с предком: если предок больше ее, это значит, что проверяемая раскраска неканонична – отсекаем ее.

Алгоритм



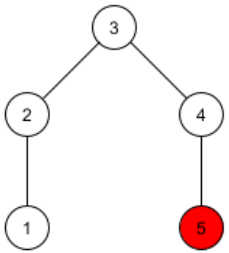
$(1, 1, 2, 1, 1)$

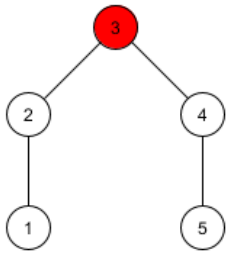
Неканоничная раскраска



$(1, 1, 2, 1, 2)$

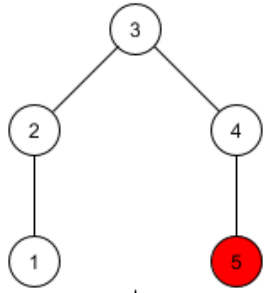
$(1, 1, 1, 1, 2)$





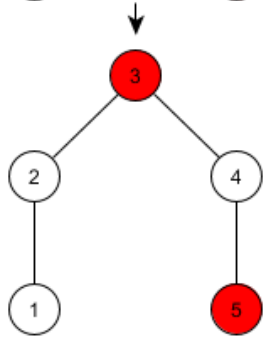
$(1, 1, 2, 1, 1)$

Алгоритм



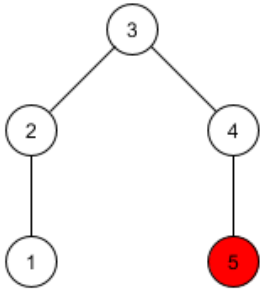
$(1, 1, 1, 1, 2)$

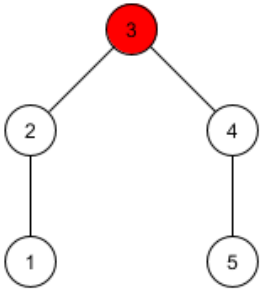
Каноничная раскраска



$(1, 1, 2, 1, 2)$

$(1, 1, 1, 1, 2)$





$(1, 1, 2, 1, 1)$

Алгоритм

Выводим все раскраски с в точности k цветами, вычисляем орбиты для каноничных раскрасок и повторяем предыдущие шаги до тех пор, пока не будут перебраны все раскраски.

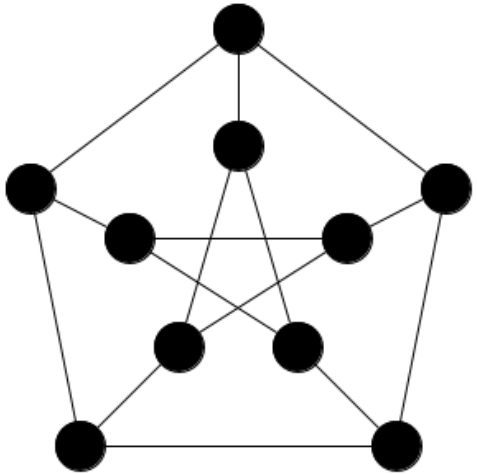
Все полученные раскраски удовлетворяют условиям лексикографической минимальности и неизоморфности.

Алгоритм

Алгоритм естественным образом переносится на случай генерации реберных раскрасок путем построения реберного графа. Подробный алгоритм построения реберного графа и генерации реберных раскрасок рассмотрен в опубликованных тезисах «*Абросимов М.Б., Разумовский П.В. О генерации неизоморфных вершинных* k *- раскрасок // Прикладная дискретная математика. Приложение. - 2017. - № 10. – С. 136-138*».

Результаты алгоритма

Граф Петерсена имеет 10 вершин и 15 ребер.



Результаты

Количество цветов (*k*)	Количество раскрасок	Время выполнения
2	247	0.0351 sec
3	8082	1.1832 sec
4	61991	17.6643 sec
5	205967	2 min 13.0289 sec
6	372133	11 min 0.6890 sec
7	381674	43 min 47.8447 sec
8	207423	2 h 28 min 53.7104 sec
9	46269	10 h 55 min 23.2296 sec
10	1	0.0000 sec

Спасибо за внимание!