

Раскраски графов

Тахонов Иван Иванович

Новосибирский государственный университет
Механико-математический факультет

НГУ, 2019

Раскраска карты (~ 1850 г.)

Каково минимальное число цветов, в которые можно раскрасить страны (регионы) на карте так, чтобы соседствующие страны (регионы) были покрашены в разные цвета?



Пусть $G = (V, E)$ – неориентированный граф и $p \in \mathbb{N}$.

Раскраской G в p цветов (**p -раскраской**) называется функция

$$\begin{aligned} c : V &\rightarrow \{1, 2, \dots, p\} && - \text{вершинная раскраска}, \\ c : E &\rightarrow \{1, 2, \dots, p\} && - \text{реберная раскраска}, \end{aligned}$$

такая, что

- (для вершинной раскраски) никакие две смежные вершины не покрашены в один цвет:

$$c(i) \neq c(j), \quad \forall (i, j) \in E,$$

- (для реберной раскраски) никакие два ребра, инцидентные одной вершине, не покрашены в один цвет:

$$c(e_i) \neq c(e_2), \quad \forall e_1, e_2 \in E : e_1 \cap e_2 \neq \emptyset.$$

Общие сведения о вершинной раскраске графа

Вершинная раскраска

Задача 1 (VERTEX COLORABILITY, p -РАСКРАШИВАЕМОСТЬ)

Дан (неор)граф $G = (V, E)$ и $p \leq |V|$.

Верно ли, что существует вершинная p -раскраска G ?

Пример: 3-РАСКРАШИВАЕМОСТЬ

Дан неорграф $G = (V, E)$.

Вопрос: $\exists?$ функция $c : V \rightarrow \{1, 2, 3\}$ такая, что $c(u) \neq c(v)$ для всех $(u, v) \in E$.

Определение 1

Минимальное число цветов, в которое можно раскрасить вершины (неор)графа G , называют **хроматическим числом G** и обозначают через $\chi(G)$.

Вопросы:

- 1 Какое множество вершин обязательно должно быть окрашено в разные цвета?

Вопросы:

- ① Какое множество вершин гарантировано окрашено в разные цвета?
- ② Как связаны хроматическое $\chi(G)$ и кликовое $\omega(G)$ числа графа? ($\omega(G)$ – макс. размер клики в G). Совпадают ли они?

Теорема 1 (Mycielski, 1955)

Пусть $k \geq 1$. Существует граф G_k : $\chi(G_k) = k$ и $\omega(G_k) \leq 2$.

Доказательство. Определим граф G_k рекуррентно:

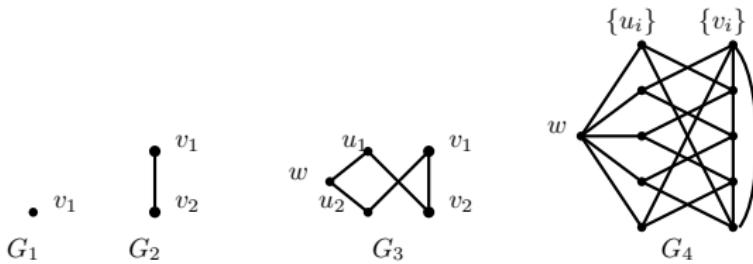
$k = 1$: G_1 – изолированная вершина.

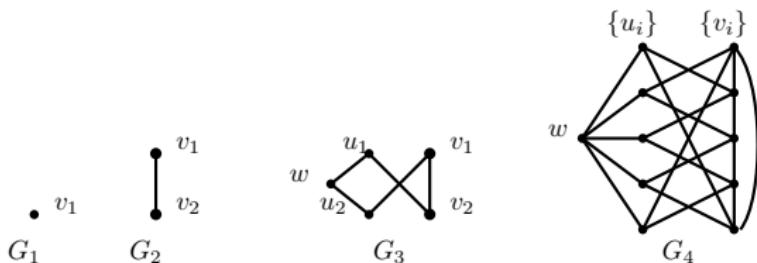
$k = 2$: G_2 – две вершины, соединенные ребром (K_2).

$k = l$: строим G_l по G_{l-1} . Пусть $V(G_{l-1}) = \{v_1, \dots, v_n\}$. Тогда

$$V(G_l) = V(G_{l-1}) \cup \{u_1, \dots, u_n\} \cup \{w\};$$

$$E(G_l) = E(G_{l-1}) \cup \{(u_i, v_j) | \forall (v_i, v_j) \in E(G_{l-1})\} \cup \{(u_i, w) | \forall u_i\}.$$



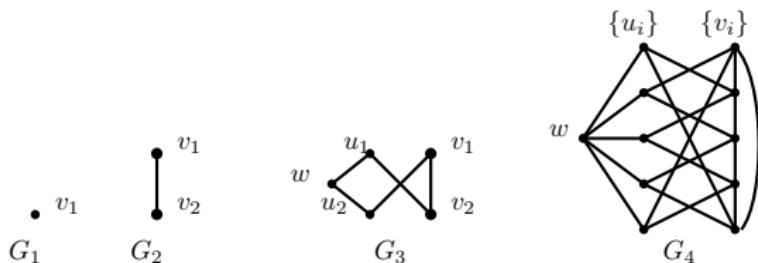


$\omega(G_k) \leq 2$: покажем, что в G_k нет треугольников. Для $k = 1, 2$ это очевидно. Пусть G_{l-1} без треугольников, докажем для G_l .

Действительно, в G_l нет треугольников, включающих только $\{v_i\}$ (иначе они были бы и в G_{l-1}).

Вершина w смежна только с $\{u_i\}$, которые не смежны между собой
 $\Rightarrow w$ тоже не входит в треугольники.

Кроме w , вершина u_i смежна с теми же вершинами, что и v_i
 \Rightarrow если бы в G_l был $\Delta u_i v_j v_h$, в G_{l-1} должен был быть $\Delta v_i v_j v_h$.



$\chi(G_k) = k$: для $k = 1, 2$ ОК. Пусть $\chi(G_{l-1}) = l - 1$. Рассмотрим G_l .

Заметим, что по $(l - 1)$ -раскраске G_{l-1} строится l -раскраска для G_l : красим u_i в тот же цвет, что и v_i , а w в цвет $l \Rightarrow \chi(G_l) \leq l$.

С другой стороны, $G_{l-1} \subset G_l \Rightarrow \chi(G_l) \geq \chi(G_{l-1}) = l - 1$. Пусть найдется раскраска G_l в $(l - 1)$ цвет. Допустим, w окрашена $l - 1$. Этот цвет не встречается среди $\{u_i\}$. Перекрасим все v_i в цвет копии u_i . Раскраска (индукционная на G_{l-1}) остается правильной, но содержит $l - 2$ цвета. ↴ Таким образом, $\chi(G_l) = l$. ■

Вопросы:

- ❶ Какое множество вершин гарантировано окрашено в разные цвета?
- ❷ Как связаны хроматическое $\chi(G)$ и кликовое $\omega(G)$ числа графа? ($\omega(G)$ – макс. размер клики в G). Совпадают ли они?
- ❸ Какое множество вершин можно красить в один цвет?

Вопросы:

- ❶ Какое множество вершин гарантировано окрашено в разные цвета?
- ❷ Как связаны хроматическое $\chi(G)$ и кликовое $\omega(G)$ числа графа? ($\omega(G)$ – макс. размер клики в G). Совпадают ли они?
- ❸ Какое множество вершин можно красить в один цвет?
- ❹ Как связаны хроматическое число $\chi(G)$ и число независимости графа $\alpha(G)$? ($\alpha(G)$ – макс. размер независимого множества G).

Замечание 1

Пусть G – простой связный граф. Верна оценка

$$\chi(G) \leq \frac{\sqrt{8|E(G)| + 1} + 1}{2}.$$

Док.-во. Рассмотрим раскраску графа в $\chi(G)$ цветов. Обозначим через $V_i \subset V$ – множество вершин цвета i :

$$V = V_1 \cup V_2 \cup \dots \cup V_{\chi(G)}.$$

Заметим:

- при $i \neq j$ м/у V_i и V_j обязательно д.б. хотя бы одно ребро;
- количество ребер м/у всеми парами V_i и $V_j \geq (\chi(G)^2 - \chi(G))/2$;
- между вершинами внутри V_i нет ребер.

Значит, $|E(G)| \geq \frac{\chi^2(G) - \chi(G)}{2}$, откуда и следует нужная оценка. ■

Таким образом, имеем следующие оценки на хроматическое число:

- ① $\chi(G) \geq \omega(G)$,
- ② $\chi(G) \geq V(G)/\alpha(G)$,
- ③ $\chi(G) \leq \frac{\sqrt{8|E(G)|+1}+1}{2}$.

Сложность задачи

Увы, задачи раскрашиваемости сложнорешаемы даже в самых рафинированных постановках..

Теорема 2

Задача 3-РАСКРАШИВАЕМОСТЬ \mathcal{NP} -полнна.

Таким образом, дать ответ на вопрос, можно ли раскрасить вершины графа в заданное количество $r \geq 3$ цветов, за полиномиальное время не получится (если $\mathcal{P} \neq \mathcal{NP}$).

Также за полиномиальное время не удастся найти хроматическое число произвольного графа и построить его вершинную раскраску в минимальное число цветов.

Для любознательных: доказательство \mathcal{NP} -полноты задачи о 3-раскрашиваемости.

Общая схема доказательства \mathcal{NP} -полноты некоторой задачи: берем другую известную \mathcal{NP} -полную задачу и сводим ее к нашей. Тем самым наша задача как бы оказывается более сложной, чем известная сложная задача. В данном случае используется NAE-3SAT – известная \mathcal{NP} -полнная задача, см. книгу: Garey, M. R.; Johnson, D. S. Computers and Intractability: A Guide to the Theory of NP-Completeness (1979)

Доказательство. Сводим Not-All-Equal 3SAT: по данной булевой функции

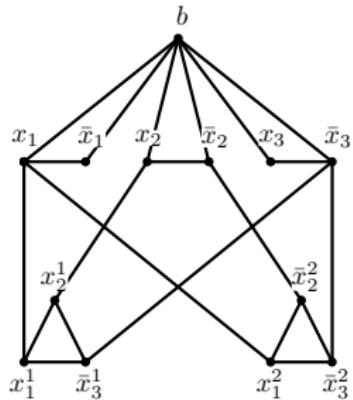
$$f(x_1, x_2, \dots, x_n) = \bigwedge_{i=1}^m C_i, |C_i| = 3 \quad \forall i,$$

сказать, если у нее not-all-equal (NAE) набор истинности (в каждой дизъюнкции есть хотя бы 1 истинный и хотя бы 1 ложный литерал).

Пусть дана б.ф. из 3SAT. Построим по ней граф $G = (V, E)$ (для 3-Раскрашиваемости) следующим образом:

- V содержит:
 - вершины x_j и \bar{x}_j для всех переменных f ;
 - вершины x_j^i (\bar{x}_j^i), если литерал x_j (\bar{x}_j) входит в дизъюнкцию C_i в f ;
 - вспомогательную вершину b .
- G содержит $\Delta x_j \bar{x}_j b$ для каждой переменной x_j .
- G содержит $\Delta y_1^i y_2^i y_3^i$ для каждой дизъюнкции $C_i = y_1^i \vee y_2^i \vee y_3^i$.
- Вершины “дизъюнктивных” треугольников соединены с одноименными “литеральными” вершинами.

$$f = (x_1 \vee x_2 \vee \bar{x}_3) \wedge (x_1 \vee \bar{x}_2 \vee \bar{x}_3)$$



Заметим, что

- есть $\Delta \Rightarrow \chi(G) \geq 3$;
- у b цвет 1 (пусть) \Rightarrow все литер. вершины окрашены в другие цвета (в 3-раскр.: 2, 3);
- x_i и \bar{x}_i окрашены по-разному,
- дизъюнкт. Δ красится \Leftrightarrow не все соотв. литературальные вершины одноцветны.

Строим набор истинности:

$$x_i = \begin{cases} 1, & \text{одноименная вершина окрашена в 2} \\ 0, & \text{одноименная вершина окрашена в 3.} \end{cases}$$

В каждой дизъюнкции есть ист. литерал $\Leftrightarrow G$ 3-раскр. ■

Вопросы:

Можно ли за полиномиальное время определить, допускает ли
граф раскраску в

- ① 1 цвет?
- ② 2 цвета?
- ③ $|V|$ цветов?

Точные и приближенные алгоритмы

Приближенные алгоритмы: жадная раскраска

Раз уж за полиномиальное время не удастся построить точное решение задачи о минимальной вершинной раскраски, попробуем найти приближенное решение.

Как насчет жадного алгоритма?

Algorithm 1 (Greedy Coloring, GC)

Все вершины не покрашены (имеют цвет 0)

for all $v \in V$ **do**

Покрасить v в мин цвет, отсутствующий у соседей.

Теорема 3

Пусть $G = (V, E)$ и $\Delta(G)$ – макс степень вершины в G .

Алгоритм GC , примененный к G ,

- использует не более $\Delta(G) + 1$ цветов,
- имеет оценку точности не более $\frac{\Delta(G)+1}{2} \leq \frac{n}{2}$.

Док.-во. $GC(G) \leq \Delta + 1$: на каждом шаге рассматриваемая вершина имеет не более Δ соседей \Rightarrow не более Δ цветов запрещены.

Оценка точности: $\chi(G) = 1$, то алгоритм использует 1 цвет.

Если $\chi(G) \geq 2$, то

$$\frac{GC(G)}{\chi(G)} \leq \frac{\Delta(G) + 1}{2}.$$

Замечание 2

Жадный алгоритм дает оценку на хроматическое число:

$\chi(G) \leq \Delta(G) + 1$, где $\Delta(G)$ – макс. степень вершины в G .

Вопросы

Достижимы ли границы? То есть:

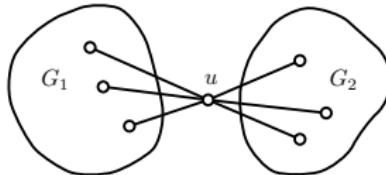
- ① есть ли график G такой, что $GC(G) = \Delta(G) + 1$?
- ② есть ли график G такой, что $\frac{GC(G)}{\chi(G)} = \frac{\Delta(G)+1}{2}$?
- ③ есть ли график G такой, что $\chi(G) = \Delta(G) + 1$?

Теорема 4 (Brooks, 1941)

Пусть G – связный граф, не являющийся полным графом или циклом нечетной длины. Тогда $\chi(G) \leq \Delta(G)$.

Доказательство. Индукцией по $|V(G)|$. Пропускаем графы с $|V(G)| = 1, 2$ (связные графы полны). При $|V(G)| = 3$ связный неполный граф – это путь, макс. степень = 2, красится в 2 цвета.

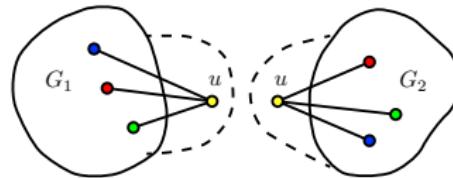
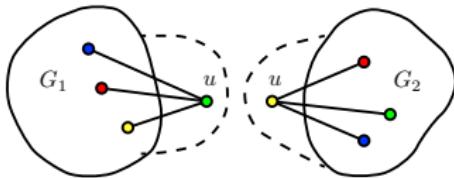
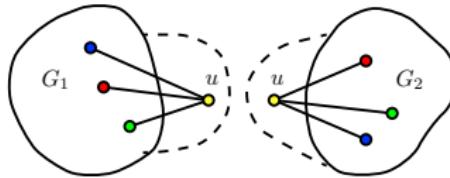
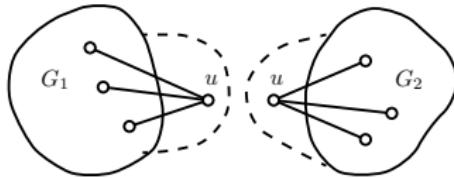
Случай 1: найдется $u \in V(G)$: $G \setminus \{u\}$ не связный ($= G_1 \cup G_2$).



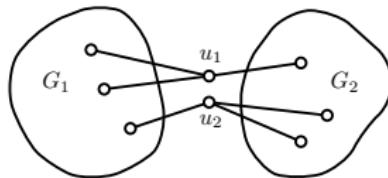
Рассмотрим подграфы $G \setminus G_i$. Каждый красится в $\leq \Delta(G)$ цветов.

Действительно, если $G \setminus G_i$ клика или нечетный цикл, то степени в нем $\leq \Delta(G) - 1$ и он красится в $\Delta(G)$. Иначе степени $\leq \Delta(G)$ и хроматическое число тоже $\leq \Delta(G)$ по индукции.

W.l.o.g. считаем, что u в обоих графах покрашена одинаково (иначе меняем цвета). Объединяя и получаем раскраску в $\leq \Delta(G)$ цветов.

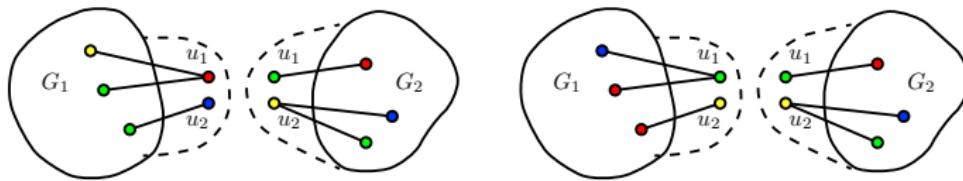


Случай 2: найдутся $u_1, u_2 \in V(G)$ такие, что $(u_1, u_2) \notin E$ и граф $G \setminus \{u_1, u_2\}$ не связный ($= G_1 \cup G_2$).

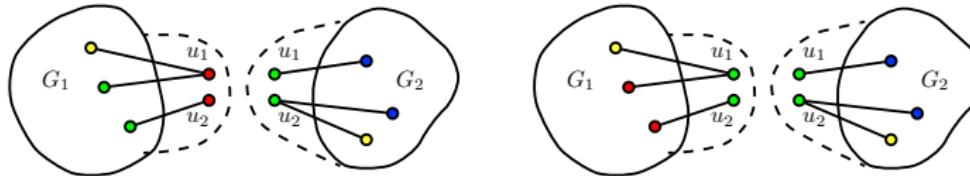


Снова рассматриваем графы $G \setminus G_i$ (как и ранее, они $\leq \Delta(G)$ -раскр.) и пробуем объединить раскраски.

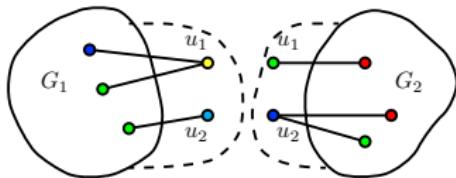
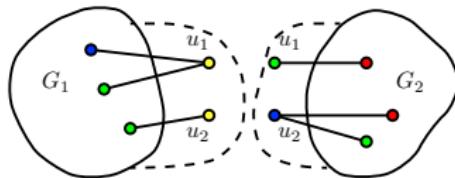
Случай 2a: в обоих графах u_1 и u_2 покрашены в два разных цвета.
Тогда можем объединить раскраски (возможно, придется поменять местами два цвета в одном из графов).



Случай 2b: в обоих графах u_1 и u_2 покрашены в два одинаковых цвета. Снова меняем цвета (при необходимости) и объединянем.



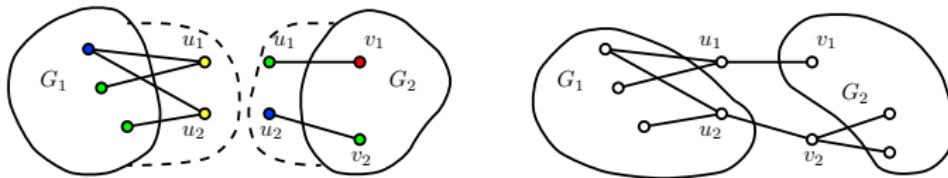
Случай 2c: в одном графе $(G \setminus G_2)$ цвета одинаковы, а в другом $(G \setminus G_1)$ различны.



Без ограничения общности считаем, что в $G \setminus G_2$ степени u_1 и u_2 одинаковы и равны $\Delta(G) - 1$.

Действительно, пусть у u_2 степень $< \Delta(G) - 1$. Тогда у нее есть свободный цвет! Перекрашиваем и получаем хороший случай (2a).

Итак, в $G \setminus G_2$ степени u_1 и u_2 одинаковы и равны $\Delta(G) - 1$. Тогда в $G \setminus G_1$ у них степень 1.

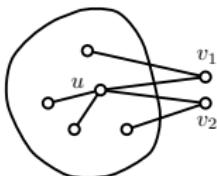


То есть, $(u_1, v_1), (u_2, v_2) \in E(G \setminus G_1)$ для некоторых v_1, v_2 . При этом:

- $v_1 \neq v_2$ – иначе Случай 1;
- $(u_1, v_2), (u_2, v_1) \notin E(G)$ – иначе степени u_1 и u_2 в $G \setminus G_1$ были бы больше 1.

Сделаем замену: $\{u_1, u_2\} \rightarrow \{u_1, v_2\}$. Получаем Случай 2, но теперь можем заменить цвета (нет проблем со степенью).

Случай 3: ни один из предыдущих случаев.

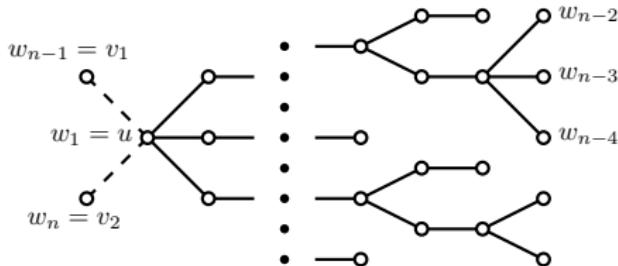


Пусть u – вершина максимальной степени. Найдутся v_1, v_2 такие, что $(u, v_1), (u, v_2) \in E$ и $(v_1, v_2) \notin E$.

Действительно, если все соседи u смежны между собой, то они (вместе с u) образуют клику C . Степень всех вершин в этой клике одинакова и равна $\Delta(G)$. Весь граф G – не клика \Rightarrow у некоторой $v \in C$ должен быть сосед $w \notin C$. Но тогда у v степень $\Delta + 1$. ∇

Итак, у вершины u макс. степени есть несмежные соседи v_1 и v_2 .

Граф $G \setminus \{v_1, v_2\}$ связен (иначе Случай 2) \Rightarrow строим остов с корнем u .

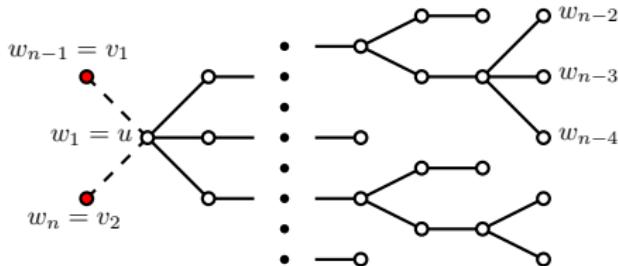


Упорядочим вершины по мере удаления от корня.

$$\{u = w_1, w_2, \dots, w_{n-2}\} \cup \{w_{n-1} = v_1, w_n = v_2\}.$$

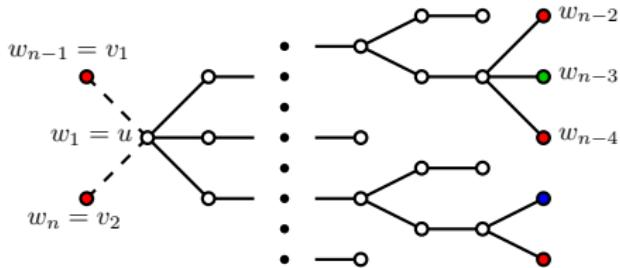
С помощью этого дерева можно покрасить G в Δ цветов!

Используем GC, перебирая вершины по убыванию номеров (по слоям дерева).



Начинаем с вершин $w_n = v_2$ и $w_{n-1} = v_1$. Они несмежны и красятся в один цвет.

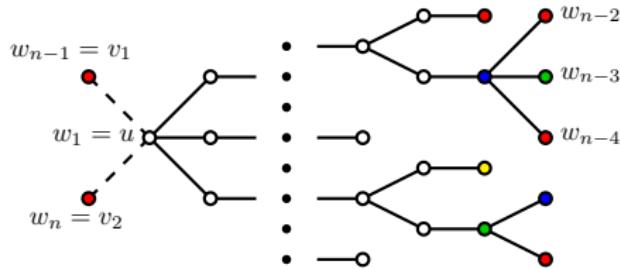
Используем GC, перебирая вершины по убыванию номеров (по слоям дерева).



Затем красим дальний слой...

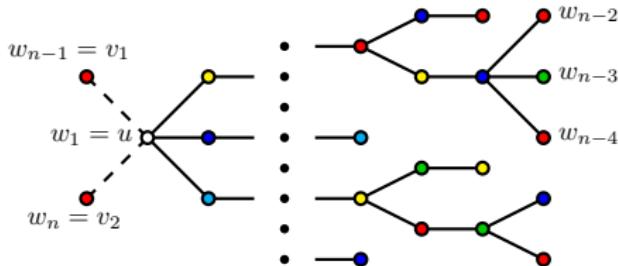
(В дереве представлены не все ребра G, так что вершины одного слоя совсем не обязательно будут выкрашены в один цвет)

Используем GC, перебирая вершины по убыванию номеров (по слоям дерева).



..предыдущий слой...

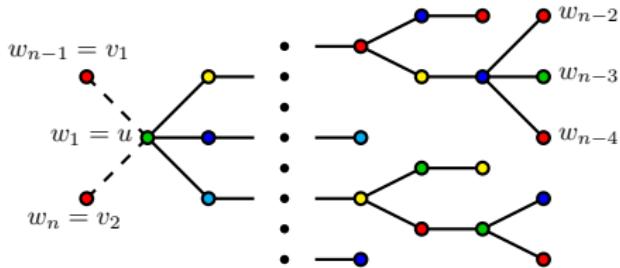
Используем GC, перебирая вершины по убыванию номеров (по слоям дерева).



..и так далее...

Заметим: на каждом из слоев кроме корневого у каждой вершины w_i есть еще не покрашенный сосед. Это ее предок. Значит, при рассмотрении w_i заняты не более $\Delta(G) - 1$ цветов, и для самой вершины есть доступная краска с номером $\leq \Delta(G)$.

Используем GC, перебирая вершины по убыванию номеров (по слоям дерева).



Как покрасить корень u ?

Степень $\Delta(G)$, некрашенных соседей нет, но есть два соседа цвета 1
 \Rightarrow есть один незанятый цвет. Используем его. Получили раскраску в $\Delta(G)$ цветов. ■

Жадная раскраска независимых множеств

Другой жадный подход: знаем, что независимое множество в графе можно красить в один цвет. Будем искать независимые множества, красить их в какой-то цвет и убирать из графа.

Как найти (некоторое) независимое множество? Можно опять использовать жадный подход: на каждом шаге выбираем вершину и затем удаляем ее из графа вместе с соседями.

Хочется построить множество побольше (в разумных пределах: за полиномиальное время наибольшее н.м. не найдем, так как задача \mathcal{NP} -трудна) \Rightarrow на каждом шаге стоит выбирать вершину минимальной степени.

Algorithm 2 (Greedy Independent Set Coloring, GIS)

procedure FINDINDSET($G = (V, E)$)

$S = \emptyset$.

repeat

Найти вершину v мин. степени в G .

$S = S \cup \{v\}$,

убрать v из G вместе с соседями

until $G = \emptyset$

return независимое множество S

Положить текущий цвет $c = 1$

repeat

$G' = G$, $S = \text{FindIndSet}(G')$

Красим S в c , затем $G = G[V \setminus S]$

$\text{Inc}(c)$.

until $G = \emptyset$

Теорема 5

Алгоритм GIS имеет оценку точности $O(n/\log n)$, $n = |V|$.

Замечание 3

Пусть $\varepsilon > 0$. Если $\mathcal{NP} \not\subseteq \mathcal{ZPP}$, не существует алгоритма, строящего $O(n^{1-\varepsilon})$ -приближенное решение задачи о вершинной раскраске с полиномиальной трудоемкостью.

\mathcal{ZPP} = Zero-error Probabilistic Polynomial time

Задачи распознавания, для которых существует вероятностная машина Тьюринга, которая

- ошибается с вероятностью 0 (с вероятностью 1 дает правильный ответ ДА/НЕТ),
- работает (в среднем) полиномиальное время,
- в худшем случае работает экспоненциальное время.

U. Feige and J. Kilian. Zero knowledge and the chromatic number // Journal of Computer and System Sciences, 57:187–199, 1998.

Точные алгоритмы: Connection–Contraction

Как найти точное решение?

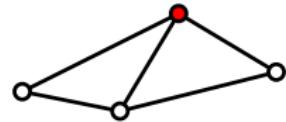
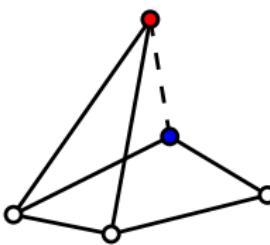
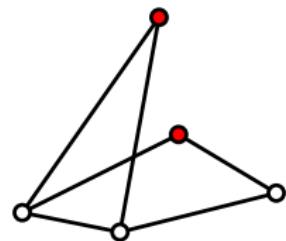
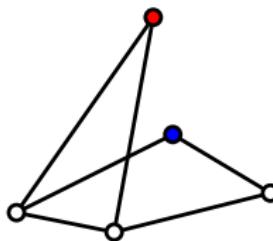
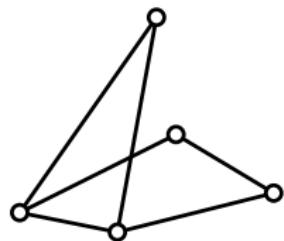
Наблюдение 1: пусть c – это \min раскраска $G = (V, E)$ и $u, v \in V$ – пара несмежных вершин, $(u, v) \notin E$.

- Если $c(u) \neq c(v)$, то c – \min раскраска для $G \cup (u, v)$.
- Если $c(u) = c(v)$, то $c|_{u=v}$ является \min раскраской $G|_{u=v}$.

Здесь

- $G|_{u=v}$ обозначает граф, в котором u и v заменены вершиной uv , смежной со всеми соседями u и v ;
- $c|_{u=v}$ обозначает раскраску $G|_{u=v}$, в которой $c|_{u=v}(x) = c(x) \forall x \neq uv$ и $c|_{u=v}(uv) = c(u) = c(v)$.

Иллюстрация



Это дает нам рекуррентные соотношения:

$$\chi(G) = \min \{ \chi(G \cup (u, v)), \chi(G|_{u=v}) \}, \quad \forall (u, v) \notin E(G)$$

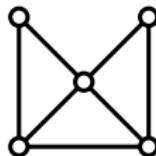
и идею алгоритма: [Connection–Contraction Algorithm](#) (Зыков, 1949).

Пусть G – неполный граф. Выбираем пару несмежных вершин и либо соединяем их, либо объединяем. Один из полученных графов имеет то же хром. число, что и исходный. Повторяем процесс, пока не получим набор полных графов. $\chi(G) =$ размеру минимального графа из набора.

Красим полный граф и возвращаемся обратно, восст. раскраску G . Если на каком-то шаге объединяли вершины, то расжимаем их обратно и красим в тот же цвет, что был у объединенной. Если добавляли ребро, то убираем его обратно.

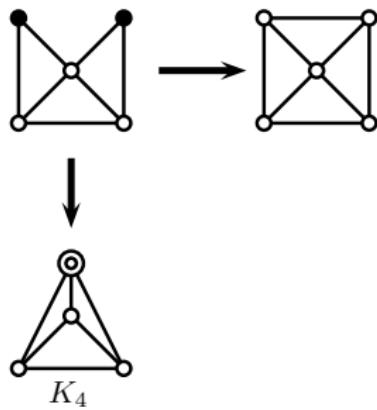
Пример

Найти мин раскраску указанного графа:



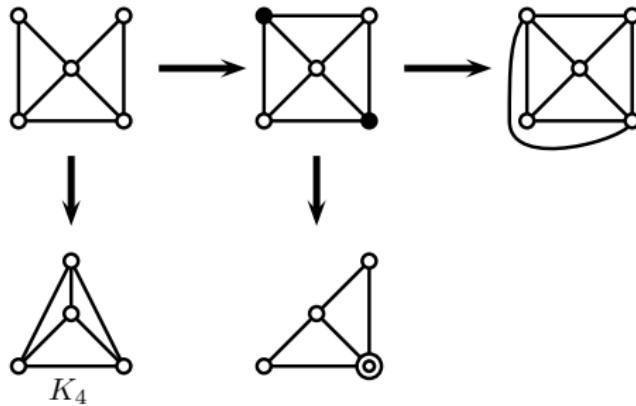
Пример

Выбираем несмежную пару, выполняем соединение и объединение. Один из полученных графов полный (K_4).



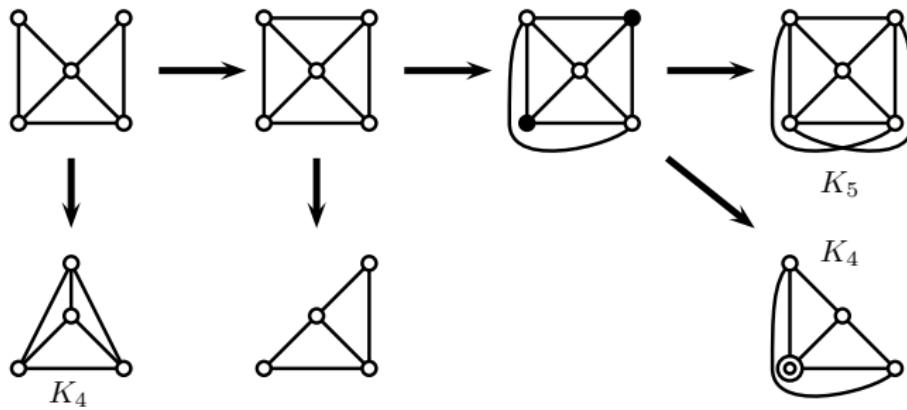
Пример

Выбираем несмежную пару во втором графе, выполняем соединение и объединение. Полных графов нет, идем дальше.



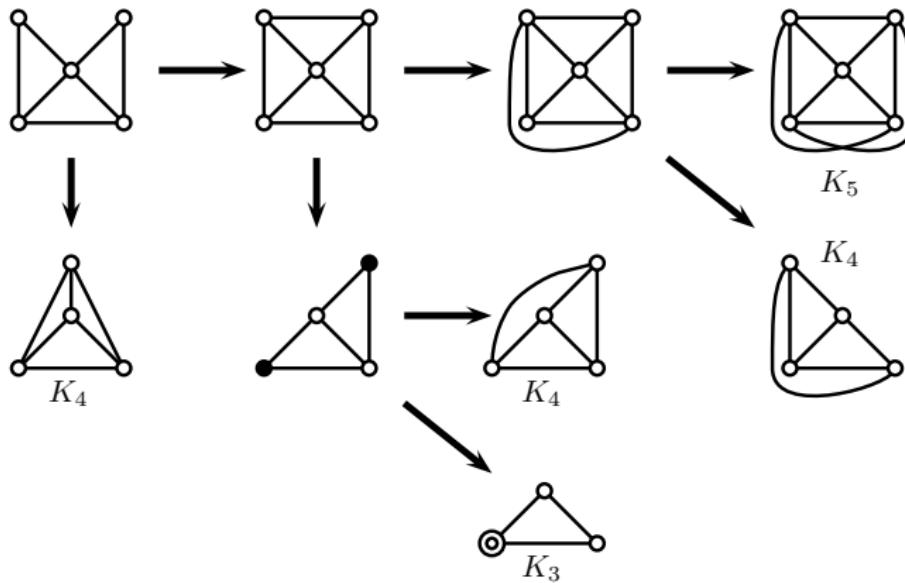
Пример

Выбираем один из графов, повторяем для него. Получили два полных графа (K_4 и K_5).



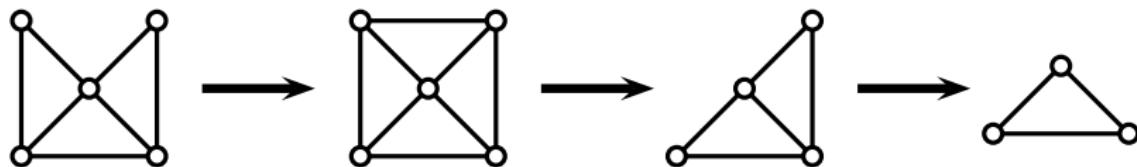
Пример

Повторяем для второго графа. Теперь все листья дерева ветвления являются полными графами, останавливаем процесс.



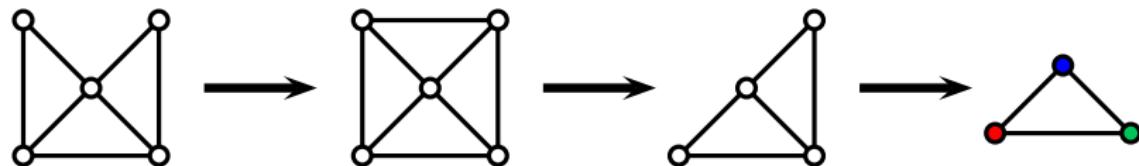
Пример

Находим ветвь, оканчивающуюся полным графом мин. размера (K_3), и восстанавливаем раскраску исходного графа.



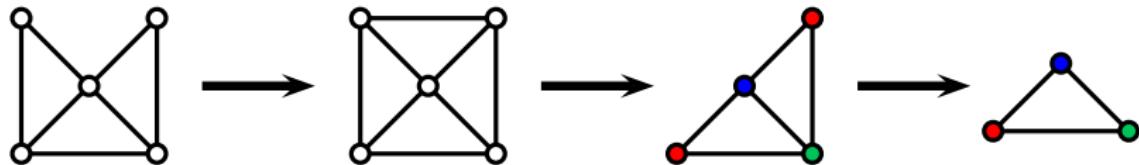
Пример

Сначала красим K_3 .



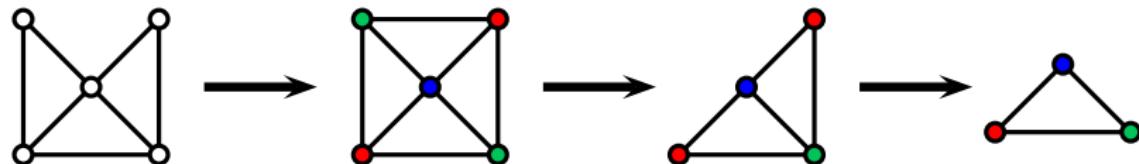
Пример

K_3 был получен объединением двух вершин. Объединенная вершина красная, так что разжимаем вершины и красим их в красный цвет.



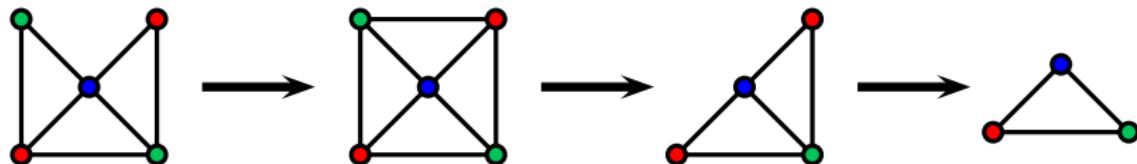
Пример

Граф получен объединением вершин. Объединенная вершина зеленая \Rightarrow разжимаем вершины и красим в зеленый цвет.



Пример

Граф получен соединением ребер. Удаляем ребро и получаем 3-раскраску исходного графа.



Заметим: этот подход можно модифицировать (наподобие метода ветвей и границ).

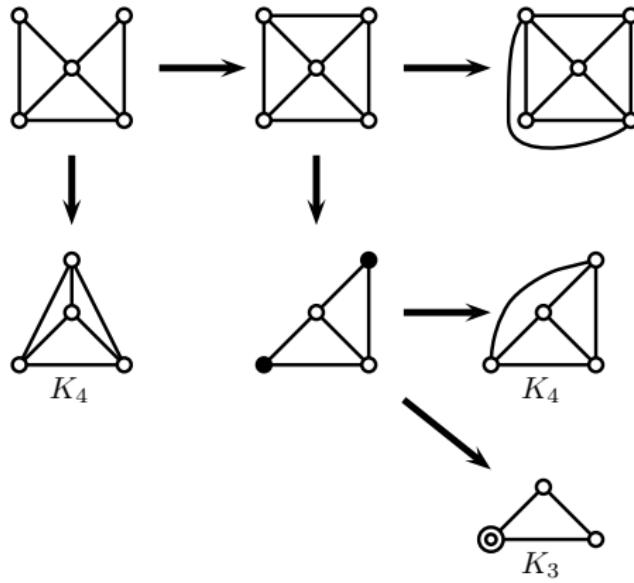
Например: D.G. Corneil and B.Graham, An Algorithm for Determining the Chromatic Number of a Graph //SIAM J. Comput., 2(4), 311–318. (1973)

Идея:

- ветвим как до этого (connection – contraction);
- нижняя граница (LB) – размер максимальной клики в текущем графе (или другая нижняя оценка на хром. число; можно использовать приближенные алгоритмы);
- верхняя граница (UB) – размер минимального полного графа, до которого доветвили на текущий момент (или верхняя оценка хроматического числа текущего графа),
- не рассматриваем графы, у которых $LB \geq UB$.

Пример

При реализации такой схемы остановили бы ветвление сразу, как дошли до K_3 – в исходном графе есть треугольники.



Точные алгоритмы: Independent Set Coloring

Наблюдение 2

- в минимальной раскраске G вершины одного цвета образуют независимое множество;
- можем считать, что одно из этих множеств **максимально по включению** (иначе перекрасим одну или несколько вершин и увеличим множество без изменения числа цветов);
- если убрать это множество (вместе и инцидентными ребрами), хроматическое число уменьшится на единицу.

Внимание: речь идет именно о максимальном по включению н.м., а не о н.м. максимальной мощности $\alpha(G)$!

Существуют графы, в которых в любой минимальной раскраске все цвета имеют мощность $< \alpha(G)$.

Это дает нам рекуррентные соотношения:

$$\chi(G) = \min_{S \in \mathcal{I}(G)} \{1 + \chi(G[V \setminus S])\},$$

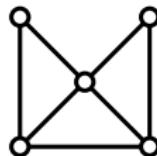
где $\mathcal{I}(G)$ – семейство макс. по вкл. независимых множеств в G , из которых следует другой простой алгоритм ([Independent Set Coloring](#), Christofides, 1971):

Пусть G – неорграф. Перебираем все макс по вкл. независимые множества $S \in \mathcal{I}(G)$ и осуществляя ветвление, поочередно убирая из графа каждое из них. Повторяем до тех пор, пока на одной из ветвей (кратчайшей) не получим граф без ребер.

Красим вершины в один цвет, затем возвращаемся обратно по ветви, добавляя на каждом шаге выброшенное множество и крася его новой краской. Рано или поздно раскрасим весь граф.

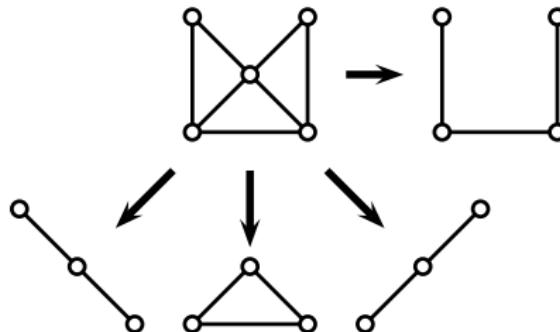
Пример

Построить минимальную раскраску указанного графа



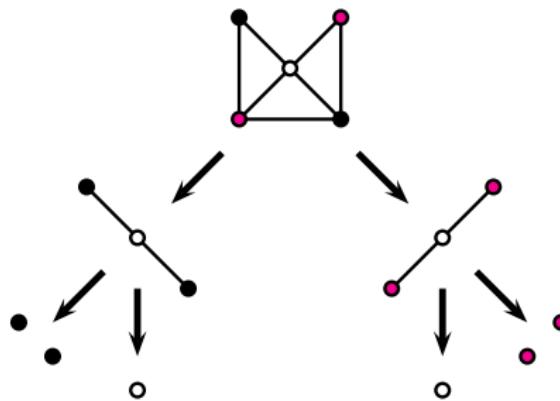
Пример

Перебираем все макс по включению независимые множества. В этом графе их три мощности 2 и одно мощности 1. Убираем каждое из них и рассматриваем полученные графы.



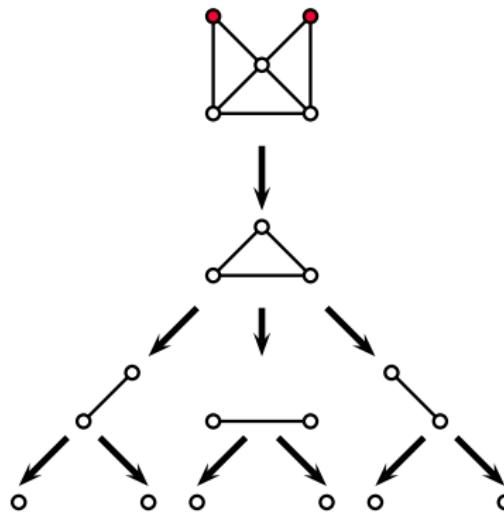
Пример

В двух из полученных графов осталось одно макс по вкл. н.м. мощности 2 и одно мощности 1. Убираем их. Каждый раз получаем графы без ребер.



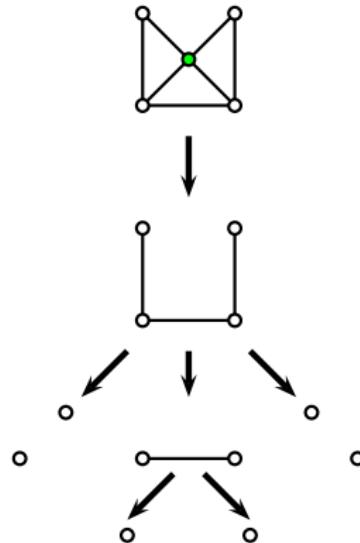
Пример

В третьем графе (треугольнике) – три независимых множества мощности два. После удаления каждого из них остается одно ребро, в котором два независимых множества мощности один.



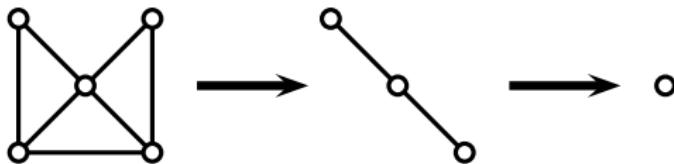
Пример

В оставшемся графе (пути) – три н.м. мощности два. После удаления двух из них остаются пустые графы, а после удаления третьего – одно ребро, в котором два н.м. мощности один.



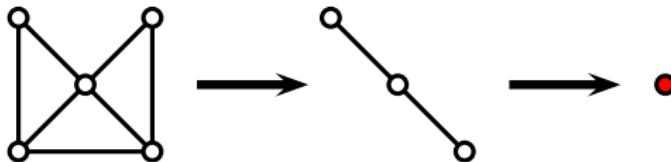
Пример

Кратчайшие ветви длины 2. Выбираем одну из них и восстанавливаем раскраску.



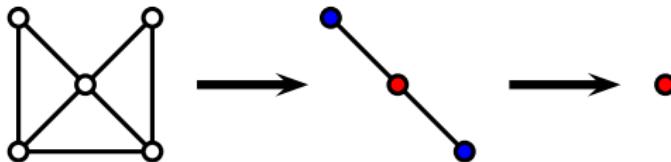
Пример

Красим вершину и восстанавливаем независимое множество.



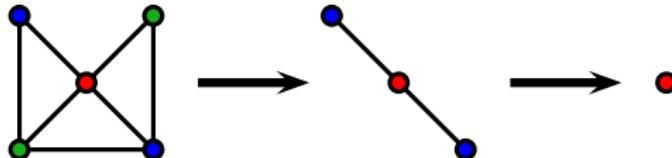
Пример

Красим множество и добавляем убранное ранее.



Пример

Красим множество. Весь граф покрашен.



Смежные вопросы: связь с расписаниями, подсчет раскрасок

Вершинные раскраски и расписания

Вершинные раскраски помогают в построении расписаний конфликтующих событий. Рассмотрим следующую задачу.

Расписание конфликтующих событий

ДАНО:

- множество событий $\{E_1, E_2, \dots, E_n\}$, каждое из которых занимает 1 единицу времени;
- для каждого события E_i известен набор событий, которые не могут проходить одновременно с ним:

$$E_i \notin \{E_{i_1}, E_{i_2}, \dots, E_{i_{d_i}}\};$$

- число доступных временных интервалов r .

ВОПРОС: можно ли провести все события за r ед. времени?



Очень похоже на r -раскраску! Строим граф конфликтов
 $G = (V, E)$:

- каждому событию сопоставляем вершину графа: $E_i \rightarrow v_i$;
- ребра соединяют конфликтующие события:

$$(v_i, v_j) \in E \Leftrightarrow E_i \not\subseteq E_j.$$

Допустимое расписание длины r существует \Leftrightarrow граф G является r -раскрашиваемым.

Заметим: верно и обратное. Задачу о раскраске вершин графа $G = (V, E)$ в p цветов можно рассматривать как задачу построения расписания:

- каждой вершине графа сопоставляем событие: $v_i \rightarrow E_i$;
- каждому ребру сопоставляем конфликт:

$$E_i \not\subset E_j \Leftrightarrow (v_i, v_j) \in E$$

Т.о., построение расписания конфликтующих событий также является сложнорешаемой задачей уже при 3 временных периодах!

Можем использовать точные и приближенные методы построения раскрасок для составления расписаний.

Пример

Рассмотрим следующий план сессии. Девять классов сдают экзамены по 8 предметам.

класс	1	2	3	4	5	6	7	8	9
экзамены	1, 2, 3	3, 6	1, 4, 6	2, 6, 8	4, 5, 8	3, 7	1, 6	2, 7	6, 7, 8

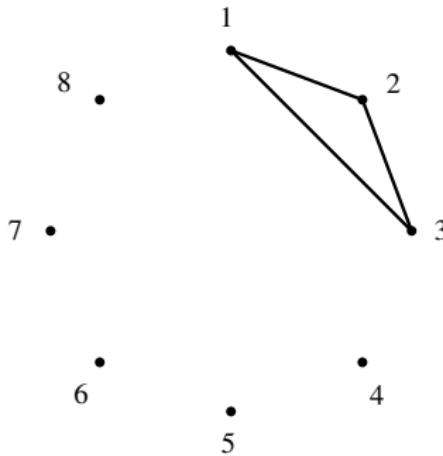
В один день каждый класс может сдавать не более одного экзамена. Все классы, сдающие один экзамен, сдают его одновременно.

- ① Определить минимальное число дней r , в которые можно провести все экзамены (и предложить расписание).
- ② Определить минимальное число аудиторий (ρ), необходимое для реализации r -дневного расписания.
- ③ Насколько больше дней потребуется, если доступна $\rho - 1$ аудитория?

Пример

Сначала построим граф, представляющий экзамены и связи между ними. Две вершины соединены ребром \Leftrightarrow у соотв. экзаменов есть общие группы.

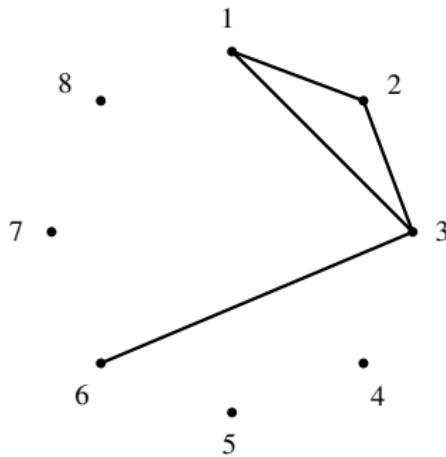
Класс 1 сдает 1, 2, 3 \Rightarrow эти три вершины образуют клику.



Пример

Сначала построим граф, представляющий экзамены и связи между ними. Две вершины соединены ребром \Leftrightarrow у соотв. экзаменов есть общие группы.

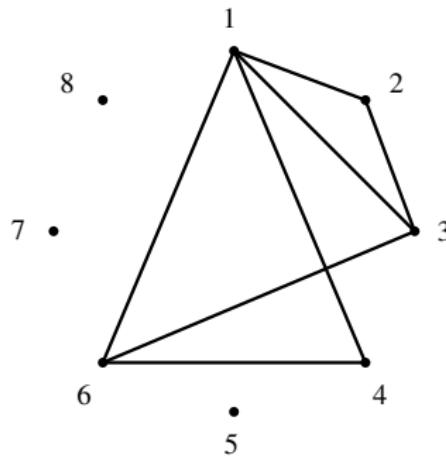
Класс 2 сдает 3, 6 \Rightarrow вершины соединены.



Пример

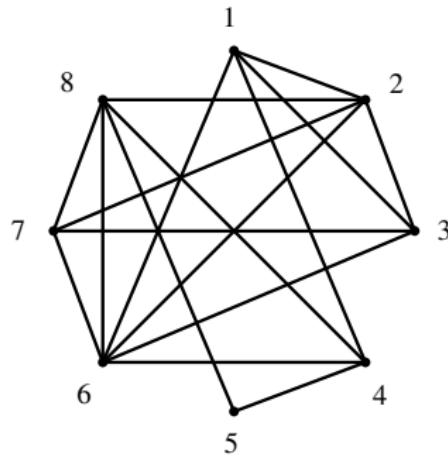
Сначала построим граф, представляющий экзамены и связи между ними. Две вершины соединены ребром \Leftrightarrow у соотв. экзаменов есть общие группы.

Класс 3 сдает 1, 4, 6 \Rightarrow еще одна клика.



Пример

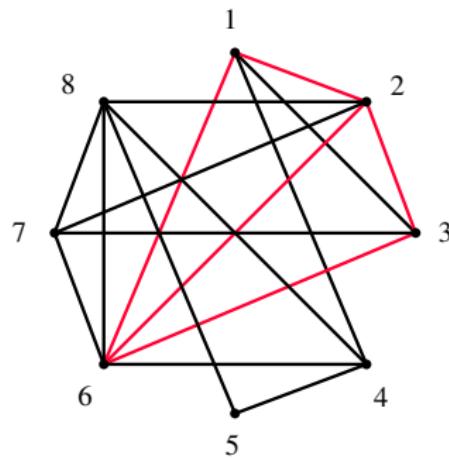
..и так далее. Получаем граф:



Ищем мин вершинную раскраску. Одноцветные вершины соотв. экзаменам, которые можно поставить одновременно.

Пример

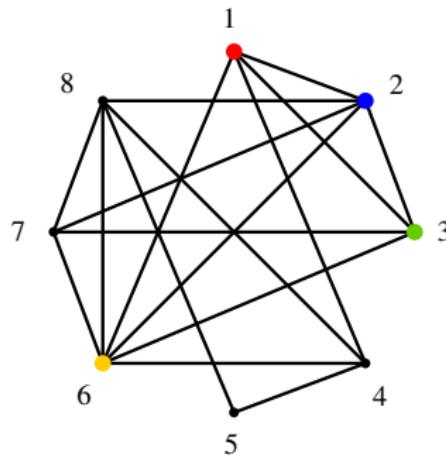
Заметим, что в графе есть клика размера 4.



Т.о., будет не менее 4 цветов.

Пример

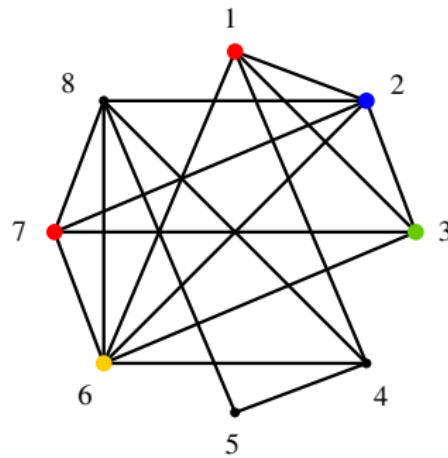
Покрасим клику в 4 цвета и попытаемся продолжить раскраску.



Вершина 7 смежна с синей, зеленой и желтой вершинами \Rightarrow
она красная

Пример

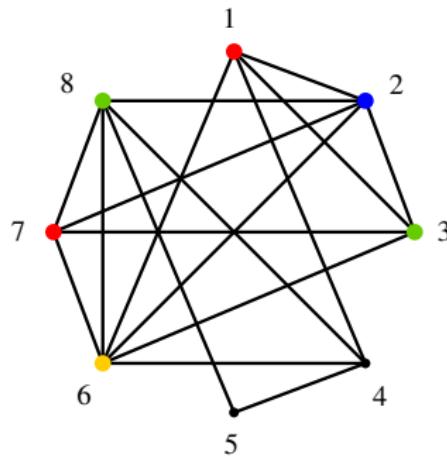
..продолжаем раскраску..



Вершина 8 смежна с синей, красной и желтой \Rightarrow она зеленая

Пример

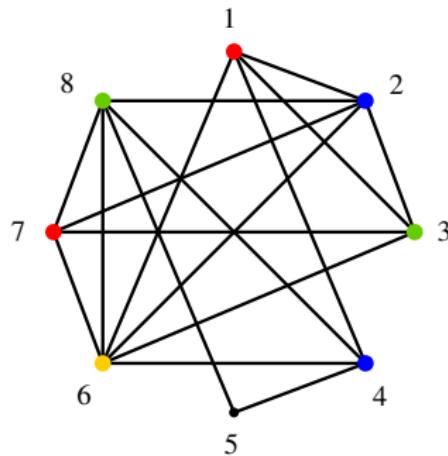
..продолжаем раскраску..



Вершина 4 смежна с зеленой, красной и желтой \Rightarrow она синяя

Пример

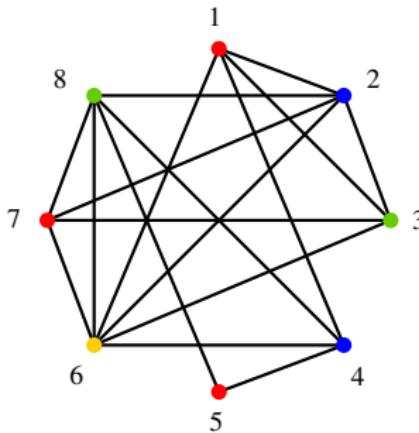
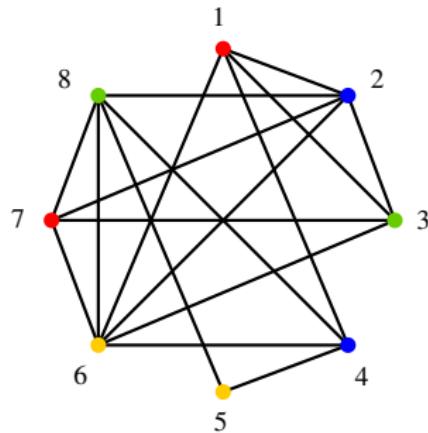
..продолжаем раскраску..



Вершина 5 смежна с зеленой и синей \Rightarrow она либо красная, либо желтая

Пример

Получили две 4-раскраски.

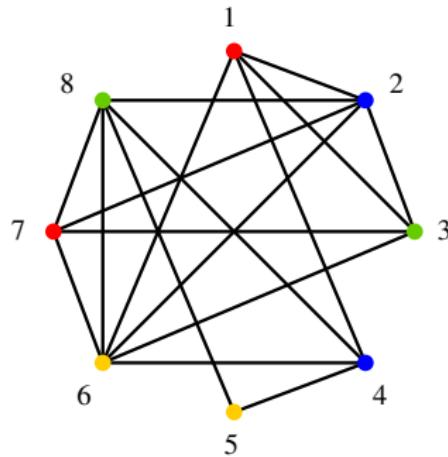


В первой все цвета имеют мощность 2 \Rightarrow нужно 2 комнаты.

Во второй раскраске есть цвет мощности 3 (красный) \Rightarrow нужна дополнительная комната.

Пример

Выбираем первую раскраску и восстанавливаем расписание
 (назначаем одноцветные экзамены в один день)



Нужно $\rho = 2$ комнаты. Если доступна $\rho - 1 = 1$ комната \Rightarrow
 допустимым является только тривиальное расписание длины 8 (один
 экзамен каждый день).

О числе вершинных раскрасок

Для любознательных

Попробуем посчитать кол.-во различных вершинных раскрасок графа G в q цветов. Обозначим эту величину через $P(G, q)$.

NB: раскраски, отличающиеся переобозначением цветов, считаем различным.

Величину $P(G, q)$ можно рассматривать как функцию переменной q для заданного графа. В этом случае $P(G, q)$ называют **хроматическим многочленом** графа G .

Для некоторых графов $P(G, q)$ вычисляется просто, например:

- график G_v , состоящий из одной вершины v

$$P(G_v, q) = q$$

- полный график K_n

$$P(K_n, q) = q \cdot (q - 1) \cdot \dots \cdot (q - (n - 1))$$

вершина 1 красится q способами, у вершины 2 – $(q - 1)$ способ, и т.д.

- P_n (простой путь длины n)

$$P(P_n, q) = q \cdot (q - 1)^{n-1}$$

вершина 1 красится q способами, для всех остальных вершин $(q - 1)$

способ, так как у них один из соседей уже покрашен

- и так далее..

Для более сложных графов посчитать число раскрасок не так легко.

Полезное наблюдение!

Пусть u, v – две несмежные вершины графа G . Добавим ребро (u, v) и посчитаем число раскрасок нового и исходного графов в q цветов.

Заметим, что число допустимых раскрасок при добавлении ребра уменьшилось: нам теперь не подходят те, в которых u и v имеют один цвет. Сколько таких раскрасок?

Ранее мы выяснили, что раскраске графа, в котором две несмежные вершины имеют один цвет, можно сопоставить раскраску графа, в котором эти вершины объединены в одну \Rightarrow число таких раскрасок равно $P(G|_{u=v}, q)$.

Получаем, если $(u, v) \notin E(G)$, то

$$P(G \cup (u, v), q) = P(G, q) - P(G|_{u=v}, q),$$

или, если считаем, что ребро $(u, v) \in E(G)$, а мы его удалили:

$$P(G, q) = P(G \setminus (u, v), q) - P(G|_{u=v}, q),$$

Другое полезное наблюдение.

Если граф G состоит из нескольких компонент связности:

$$G = G_1 \cup G_2 \cup \dots \cup G_k, \text{ то}$$

$$P(G, q) = P(G_1, q) \cdot P(G_2, q) \cdot \dots \cdot P(G_k, q).$$

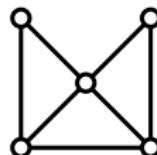
(Очевидно)

Сделанные наблюдения подсказывают, как можно найти число раскрасок произвольного графа в q цветов: [действуем рекуррентно!](#)

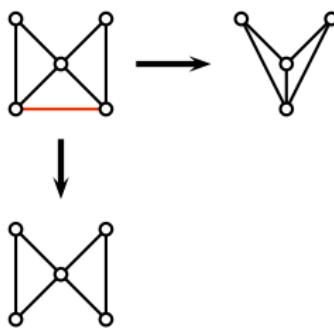
- Выбираем некоторое ребро $(u, v) \in E(G)$ и рассматриваем графы $G \setminus (u, v)$ и $G|_{u=v}$ (потом значения $P(\cdot, q)$ нужно будет вычесть одно из другого).
- Если G при удалении ребра стал несвязным, рассматриваем каждую компоненту связности отдельно (потом значения $P(\cdot, q)$ нужно будет перемножить).
- Выбираем какое-нибудь ребро в каждом из полученных графов и повторяем удаление/стягивание до тех пор, пока не получим графы, для которых можем посчитать $P(\cdot, q)$: полные графы, пути, изолированные вершины и т.п.
- Вычисляем $P(G, q)$, поднимаясь по дереву ветвления и перемножая/вычитая $P(\cdot, q)$.

Пример

Найдем число q -раскрасок указанного графа:



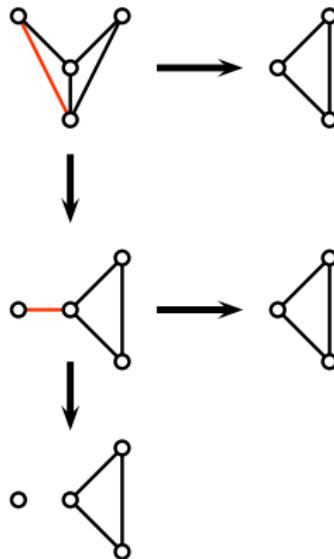
Пример



Удалим и стянем красное ребро. Получим два графа.
Рассмотрим их по отдельности.

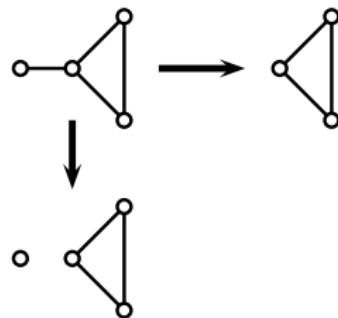
Пример

Первый граф

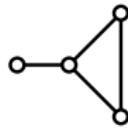


Дошли до полных графов и изолированных вершин.

Пример

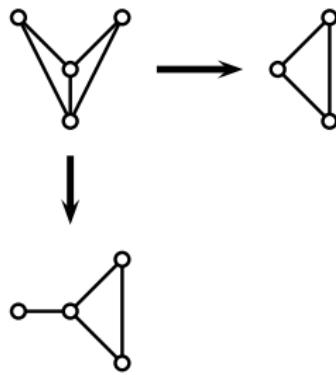


Значит

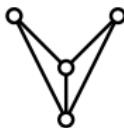


$$q \cdot q(q-1)(q-2) - q(q-1)(q-2) = q(q-1)^2(q-2)$$

Пример



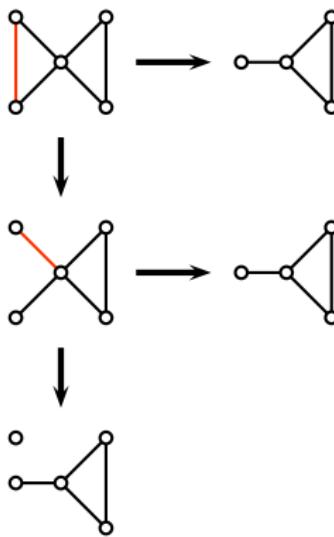
Значит



$$q(q-1)^2(q-2) - q(q-1)(q-2) = q(q-1)(q-2)^2$$

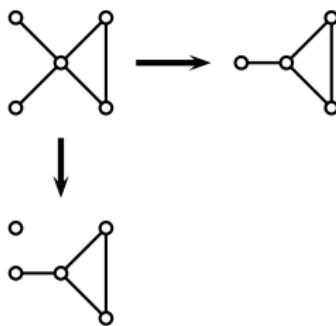
Пример

Второй граф

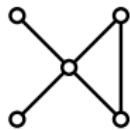


Дошли до известных графов

Пример

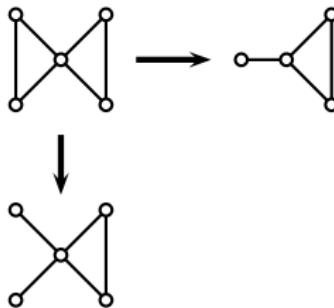


Значит

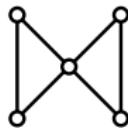


$$q \cdot q(q-1)^2(q-2) - q(q-1)^2(q-2) = q(q-1)^3(q-2)$$

Пример



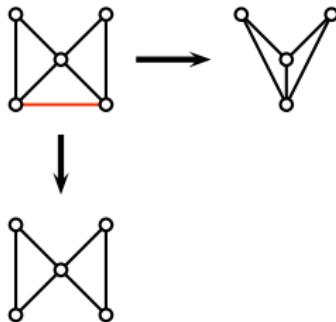
Значит



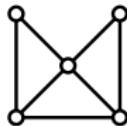
$$q(q-1)^3(q-2) - q(q-1)^2(q-2) = q(q-1)^2(q-2)^2$$

Пример

Возвращаемся к исходному графу



Значит



$$q(q-1)^2(q-2)^2 - q(q-1)(q-2)^2 = q(q-1)(q-2)^3$$

Замечания

Можем проводить вычисления, используя соотношения в виде

$$P(G, q) = P(G \cup (u, v), q) + P(G|_{u=v}, q),$$

где u, v – пара несмежных вершин. В таком случае ветвление будет проходить как в описанном ранее Connection–Contraction алгоритме и в конце нужно будет сложить значения функции для всех получившихся полных графов.

Для графа из примера ветвление уже проделывали. В итоге получалось: $1 \times K_5$, $3 \times K_4$ и $1 \times K_3$. Значит,

$$\begin{aligned} P(G, q) &= P(K_5, q) + 3P(K_4, q) + P(K_3, q) \\ &= q(q-1)(q-2)(q-3)(q-4) + 3q(q-1)(q-2)(q-3) \\ &\quad + q(q-1)(q-2) \\ &= q(q-1)(q-2)[q^2 - 4q + 4] = q(q-1)(q-2)^3. \end{aligned}$$

Замечания

Пусть G – некоторый граф с хроматическим числом χ .

Несложно понять, что

$$P(G, q) = 0 \text{ для } q = 0, 1, \dots, \chi - 1, \text{ и } P(G, \chi) > 0.$$

Значит, если бы мы умели вычислять хром. многочлен или находить его значения в целочисленных точках, то смогли бы найти хроматическое число графа.

Вывод: эффективно (за полиномиальное время) вычислять хроматические многочлены скорее всего нельзя.