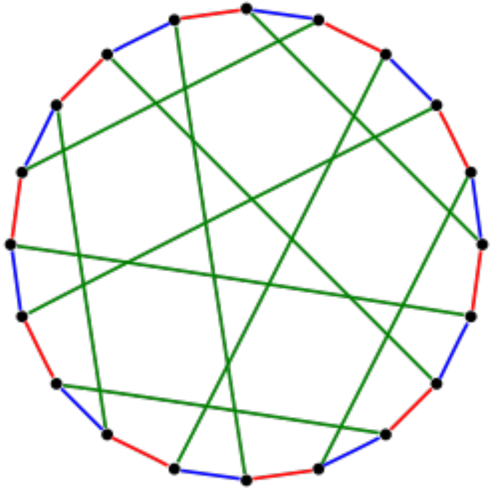
Edge coloring problem: выбор алгоритма

Автор: Митенев Алексей, СПБГУ, математико-механический факультет, 371 группа

Описание проблемы

- Назначение «цветов» рёбрам графа таким образом, что никакие два смежных ребра не имеют один и тот же цвет.
- Использовать наименьшее количество цветов



Алгоритмы раскраски

- Для особенных типов графов (двудольные, полные и т.д.)
- Раскраска большим количеством цветов
- Точные

Жадный алгоритм

- Раскраска произвольного графа максимум 2Δ - 1
- Сложность m * (2Δ - 1)
- Онлайн алгоритм

Теорема Визинга

- Рёбра любого неориентированного графа могут быть раскрашены в число цветов, максимум на единицу большего максимальной степени вершин Δ графа.
- При этом, любой граф можно раскрасить либо в Δ, либо в $\Delta + 1$ цвет

Раскраска Δ + 1 цветами

- Раскраска произвольного графа максимум Δ + 1 цветами, где Δ - максимальная степень вершины
- Сложность: $O(nm + \Delta^2 m)$; n- |V(G)|; m - |E(G)|
- Алгоритм описан в статье J. Mishra, D. Gries A constructive proof of Vizing's Theorem

Определения

G - граф, $v_0 \in V(G)$ - вершина, c_0, c_1 - цвета.

- Цвет c_0 отсутствует в v_0, если ни одно ребро, инцидентное v, не раскрашено в c_0.
- Подграф $G(c_0, c_1)$ - Подграф G, содержащий только ребра, окрашенные в c_0 или c_1.

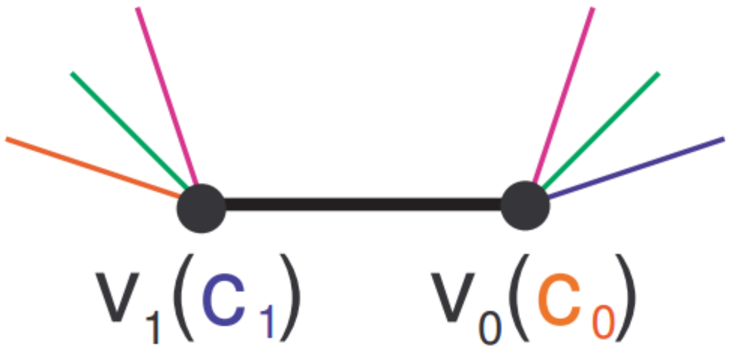
Описание алгоритма

Раскраска произвольного графа максимум $\Delta + 1$ цветами, где Δ - максимальная степень вершины

Алгоритм описан в статье J. Mishra, D. Gries A constructive proof of Vizing's Theorem

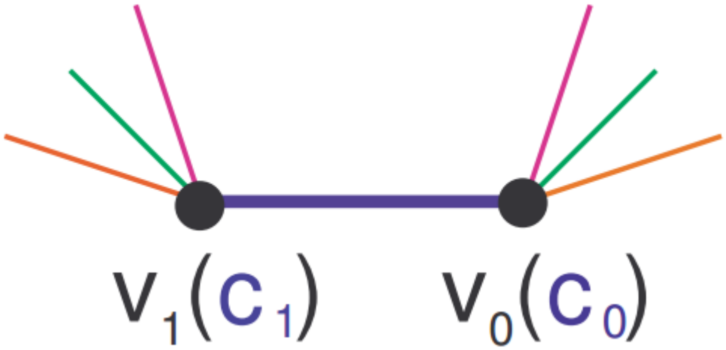
Алгоритм

- Раскрасим ребра, следующие в произвольном порядке, в цвета $\{1,2,...,\Delta+1\}$.
- Пусть (v_0,v_1) - следующее ребро для окраски
- Максимум $\Delta-1$ ребер, инцидентных v_0 или v_1 раскрашены => Существуют цвета c_0, c_1: c_0 отсутствует в v_0, c_1 отсутствует в v_1.



$c_1 = c_0$

Если $c_1 = c_0$ => Красим ребро в этот цвет



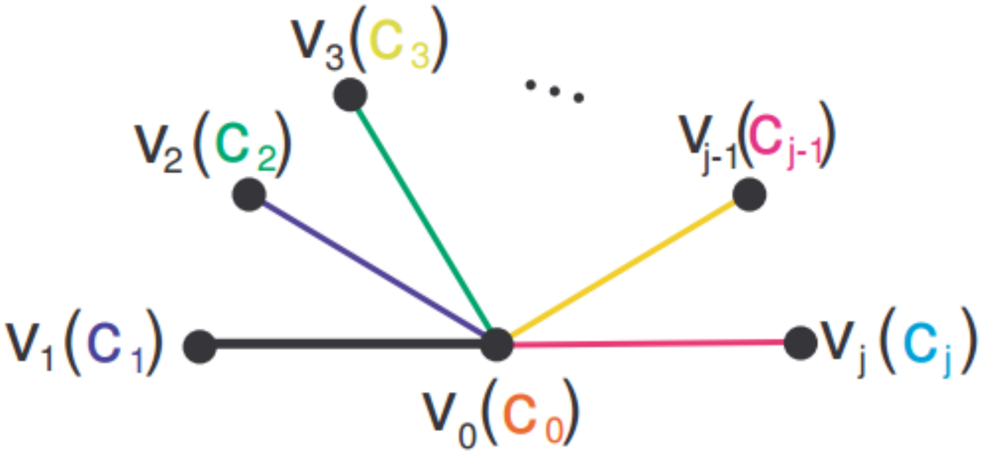
$c_0 \mathrel{!=} c_1$

Составляем последовательность цветов c_0, c_1, c_2, … ,c_{j-1} cj

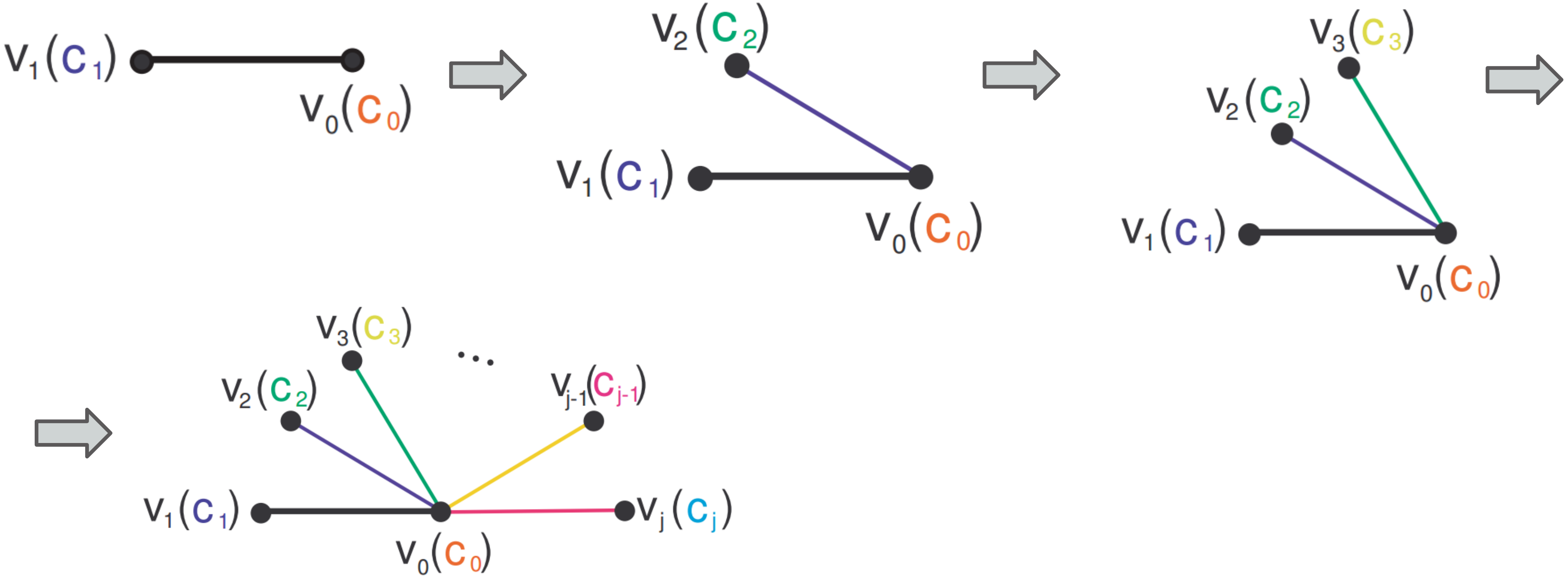
Составляем последовательность ребер (v_0,v_1) (v_0,v_2) .. (v_0, v_j)

Таким образом, что:

- цвет c_i отсутствует в v_i
- c_i - цвет ребра (v_0, v_i+1)



Составление последовательности



v_0 имеет максимум Δ соседних вершин, поэтому процесс завершится

Завершение построения

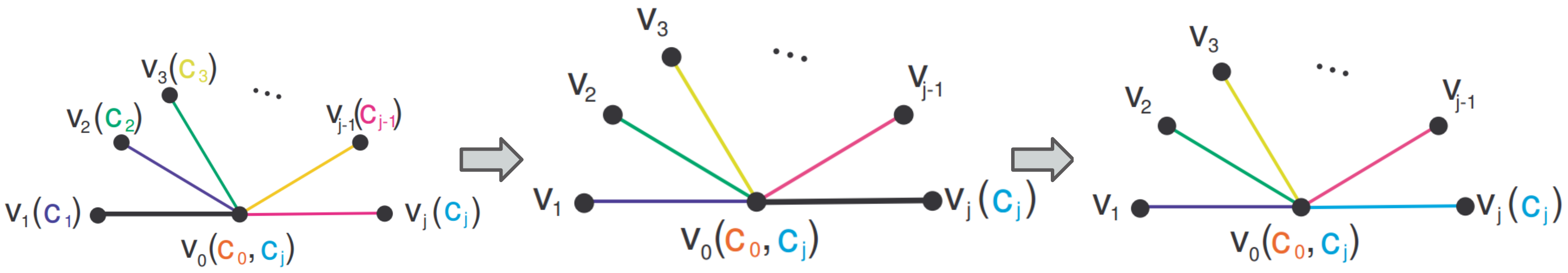
Возможны 2 случая:

1. Нет ни одного (v_0,v_i) с цветом c_j
2. Такое ребро имеется

Случай 1

Цвет c_j отсутствует в v_0

1. Тогда: сдвигаем цвета - ребро (v_0,v_i) красим в c_i, $0 < i < j$
2. c_j отсутствует и в v_j и в v_0 => красим ребро (v_0,v_j) в c_j

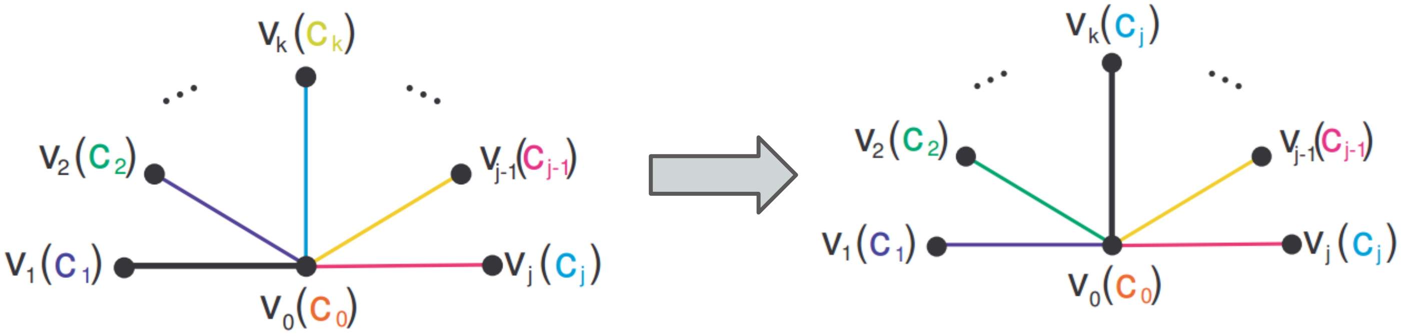


Случай 2

Какое то из ребер (v_0,v_k) имеет цвет c_j

Тогда сдвигаем цвета у ребер (v_0, v_i), $0 < i < k$

Теперь (v_0,v_k) не окрашено и c_j отсутствует в v_k



Случай 2

$G' = G(c_0, c_j)$;

$v_0, v_k, v_j \in V(G')$;

В G' любая вершина имеет одну или 2 соседние вершины.

v_0, v_k, v_j имеют только одну соседнюю вершину по построению

Таким образом, v_0, v_k, v_j не могут находиться в одной компоненте связности графа G'.

Рассмотрим:

Случай 2.1 - v_0 и v_k в разных компонентах связности G'

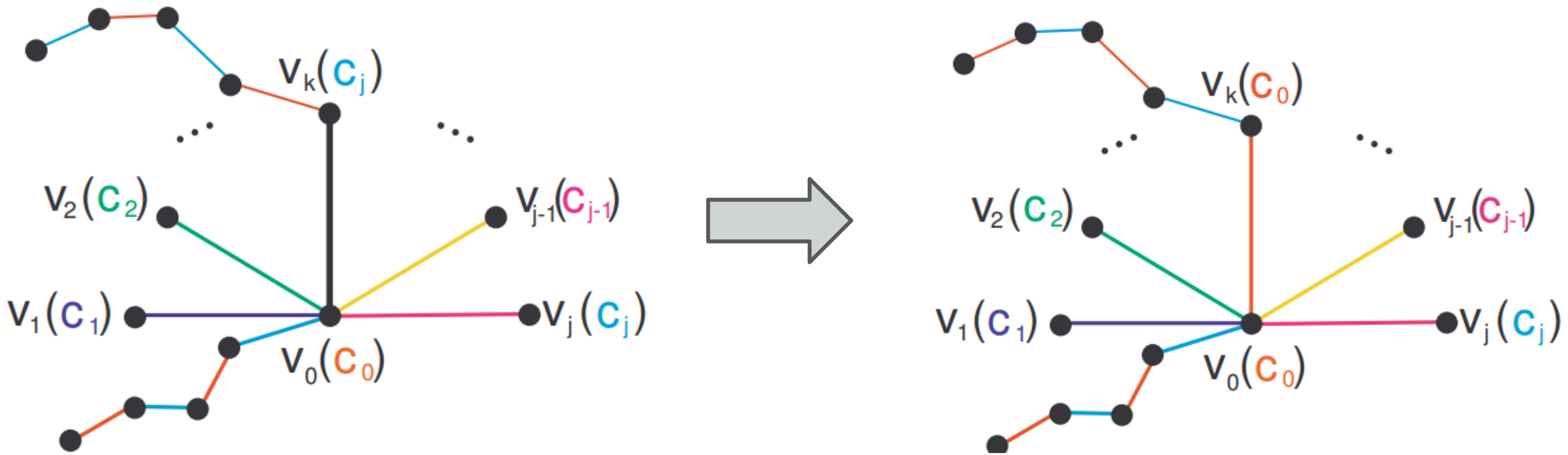
Случай 2.2 - v_0 и v_j в разных компонентах связности G'

Случай 2.1

v_0 и v_k в разных компонентах связности G’

В компоненте связности G'_k, к которой принадлежит v_k, Меняем цвет у каждого ребра c_0 => c_j, c_j => c_0

Теперь c_0 отсутствует у v_k, красим (v_o, v_k) в c_0

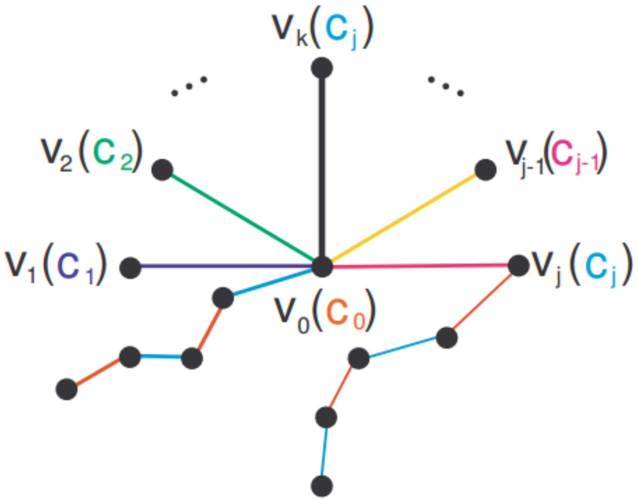


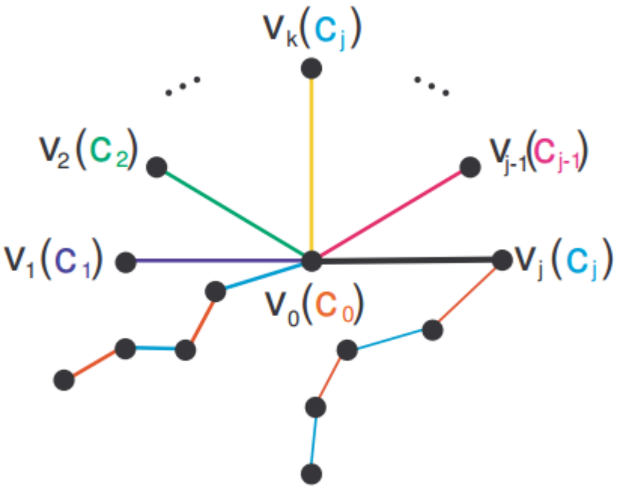
Случай 2.2

v_0 и v_j в разных компонентах связности G’;

Сдвигаем цвета у ребер (v_0,v_i), $k <= i < j$, т.е. (v_0,v_i) красим в c_i

Теперь v_j не окрашено.



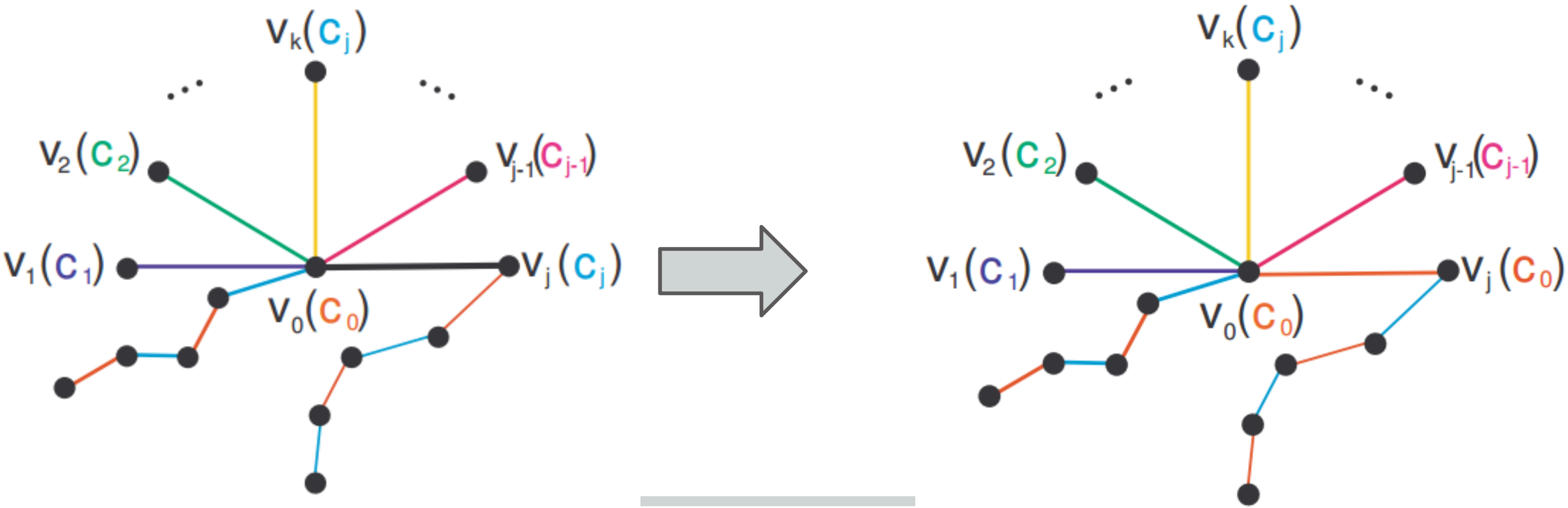


Случай 2.2

Процесс сдвига не затронул (c0,cj),

таким образом, v0, vj все еще в разных компонентах связности.

Теперь аналогично случаю 2.1



Обоснование выбора

- Алгоритм приближенный
- Раскраска максимум в Δ+1 цвет - достаточно хорошее приближение