# Вершинная 3-раскрашиваемость графа

Дмитрий Камальдинов

## 1 Введение

Целью данного проекта является имлементация алгоритмов проверки 3-раскрашиваемости графов.

**Определение 1.** *Правильной раскраской* графа $G = G(V, E)$ в $k$ цветов называется такое разбиение $V = V_1 \sqcup V_2 \sqcup \dots \sqcup V_k$ множества его вершин на $k$ подмножеств, что $\forall e \in V_i \times V_j : e \notin E \quad (i \neq j)$.

**Определение 2.** Граф называется *$k$–раскрашиваемым*, если для него существует правильная раскраска в $k$ цветов.

**Определение 3.** Пусть имеются множество $V$ из $n$ объектов и множество $C$ из $k$ *цветов*. Множество цветов, в которые может быть раскрашен объект $v \in V$ обозначим через $c(v)$. Также имеется множество *ограничений* $W \subset 2^{(V \times C)}$. *Задача удовлетворения ограничений (CSP, constraint satisfaction problem)* заключается в отыскании такой раскраски $\alpha : V \to C$, что

- $\forall v \in V \alpha(v) \in c(v)$
- $\forall w \in W \exists (v, c) \in w : \alpha(v) \neq c$.

**Определение 4.** (n, m)–CSP — такая задача CSP, что

- $|C| = n$
- $\forall w \in W : |w| = m$

**Лемма 1.** *Задача $k$–раскрашиваемости сводится к (k,2)–CSP.*

*Доказательство.* Обозначим данный граф через $G$. Следующая задача CSP эквивалентна задаче $k$–раскрашиваемости $G$:

- $V = G(V)$
- $C = \{1, \dots, k\}$
- $c(v) = C$
- $W = \{\{(v, c), (u, c)\} | \{v, u\} \in E(G), c \in C\}$

□

## 2 Описание алгоритма решения (3,2)–CSP

Определим *порядок* задачи CSP как мощность множества объектов. Объекты будут также называться *вершинами*.

**Лемма 2.** *Если в задаче (3,2)–CSP порядка $n$ существует объект, который может быть покрашен в не более, чем 2 цвета, то эта задача эквивалентна некоторой порядка $n - 1$.*

*Доказательство.* Обозначим этот объект через $v$. Рассмотрим 3 случая.

1. $|c(v)| = 0$. Тогда правильной раскраски не существует, то есть задача уже решена.

2. $|c(v)| = 1$. Обозначим единственный доступный цвет $v$ через $c$. Тогда если задача имеет решение, то $v$ обязана быть покрашеной в $c$. Значит, для каждого ограничения $\{(v, c), (u, \widehat{c})\}$, у вершины $u$ можно запретить цвет $\widehat{c}$, после чего удалить вершину $v$. Полученная задача эквивалентна исходной с учетом уже выбранного цвета для $v$, а её порядок на 1 меньше.

3. $|c(v)| = 2$. Обозначим через $c_1, c_2$ доступные цвета для $v$. $C_i \coloneqq \{(u, c) | \{(v, c_i), (u, c)\} \in W\}$. Добавим ограничения $\{(u, c), (w, \widehat{c})\} \quad \forall (u, c) \in C_1, (w, \widehat{c}) \in C_2$ и удалим вершину $v$. Тогда полученная задача эквивалентна исходной. Действительно, пусть $\alpha$ — решение (раскраска) исходной задачи. Не умаляя общности, предположим, $\alpha(v) = c_1 \implies \forall (u, c) \in C_1 : \alpha(u) \neq c$, следовательно, все новые ограничения также будут удовлетворены. Тогда $\alpha|_{V \setminus \{v\}}$ — решение новой. Напротив, пусть $\beta$ — решение новой задачи. Покрасим $V \setminus \{v\}$ с помощью $\beta$. Предположим противное: невозможно покрасить $v$ в один из цветов $c_1, c_2$ так, чтобы получилась правильная покраска вершин исходной задачи. Тогда существуют такие $(u, c)$ и $(w, \widehat{c})$, что $\{(u, c), (v, c_1)\} \in W$, $\{(w, \widehat{c}), (v, c_2)\} \in W$, и $\beta(u) = c$, $\beta(w) = \widehat{c}$. Но тогда $(u, c) \in C_1$ и $(w, \widehat{c}) \in C_2$, и одно из добавленных ограничений не удовлетворено.

□

**Лемма 3.** *Дана задача (3,2)–CSP порядка $n$, имеющая решение. Предположим, существуют два таких объекта $v$ и $u$, что $|c(v)| = |c(u)| = 3$ и $\exists c_u, c_v : \{(v, c_v), (u, c_u)\} \in W$. Тогда эта задача эквивалентна некоторой порядка $n - 2$.*

*Доказательство.* Изменим исходную задачу четырьмя способами: в каждом сохраним все ограничения, но оставим для $v$ и $u$ только по 2 возможных цвета:

1. $c(v) = \{c_v, c_v + 1\}; \quad c(u) = \{c_u + 1, c_u + 2\}$

2. $c(v) = \{c_v, c_v + 2\}; \quad c(u) = \{c_u + 1, c_u + 2\}$

3. $c(v) = \{c_v + 1, c_v + 2\}; \quad c(u) = \{c_u, c_u + 1\}$

4. $c(v) = \{c_v + 1, c_v + 2\}; \quad c(u) = \{c_u, c_u + 2\}$ (везде сложение по модулю 3)

Пусть одна из этих задач имеет решение $\beta$. Тогда $\beta$ также есть решение исходной задачи, так как в каждом случае невозможно покрасить $v$ и $u$ одновременно в $c_v$ и $c_u$. Напротив, пусть $\alpha$ — решение исходной задачи, и $\alpha(v) = \widehat{c}_v, \alpha(u) = \widehat{c}_u$, где $(\widehat{c}_v, \widehat{c}_u) \neq (c_v, c_u)$. Несложным перебором можно убедиться, что в **двух**(!) из изменённых задач $c(v) \ni \widehat{c}_v$ и $c(u) \ni \widehat{c}_u$. Тогда $\alpha$ есть решение этих задач.

Для завершения доказательства осталось избавиться от этих двух вершин с двумя доступными цветами, используя лемму 2. □

**Лемма 4.** *Дана задача (3,2)–CSP $I$ порядка $n$. Существуют два таких объекта $v$ и $u$, что $|c(v)| = |c(u)| = 3$ и $\exists c_u, c_v : \{(v, c_v), (u, c_u)\} \in W$. Тогда её можно свести за полиномиальное время к такой $I'$, что если $I'$ разрешима, то и $I$ имеет решение, а если $I$ разрешима, то $I'$ имеет решение с вероятностью $1/2$.*

*Доказательство.* Очевидно следует из леммы 3. □

Используя лемму 4, несложно убедиться в корректности следующего алгоритма ($n$ — количество вершин):

```
function SATISFIABLE(I)
    for i = 1..2^(n/2) do
        if ∃v ∈ V : |c(v)| = 3 and v has any constraints then
            reduce I → I' randomly using lemma 4
            return satisfiable(I')
        else if ∃v ∈ V : |c(v)| = 0 then
            return false
        else
            choose any v (now 0 < |c(v)| ≤ 2 and reduce I → I' using lemma 2
            return satisfible(I')
        end if
    end for
    return false
end function
```

Действительно, если $n/2$ раз случайно выбирать вариант меньшей задачи, то, согласно лемме 4, вероятность получить правильную раскраску есть $2^{-n/2}$, то есть если вызвать алгоритм $2^{n/2}$ раз, то верный ответ будет получен с вероятностью 1. Итого время работы этого алгоритма $O(O(1)2^{n/2}) = O(1.4142135623730951^n)$

#TODO:

- более подробным разбором случаев можно улучшить константу до 1.38028
- в случае задачи 3–раскрашиваемости константа (3,2)–CSP улучшается до 1.3645 с помощью некоторого препроцессинга на графе

# 3 Эксперименты

- Алгоритм работает корректно на тестах (проверены около 100 случайно сгенерерованных графов заведомо 3–раскрашиваемых/нераскрашиваемых)
- Алгоритм тратит на подтверждение 3–раскрашиваемости графа на 100 вершинах до 16 секунд, в среднем 3 (плотность 0.6).
- До 14 секунд на нераскрашиваемых на 35 вершинах (плотность 0.5)
- Алгоритм работает дольше на графах с меньшей плостностью

# Список литературы

[1] Richard Beigel, David Eppstein, *3–Coloring in time $O(1.3446^n)$: a no–MIS algorithm*, 36th Symposium on Foundations of Computer Science, 444–453, October 1995

[2] David Eppstein, *Improved algorithms for 3–Coloring, 3–Edge–Coloring, and Constraint Satisfaction*