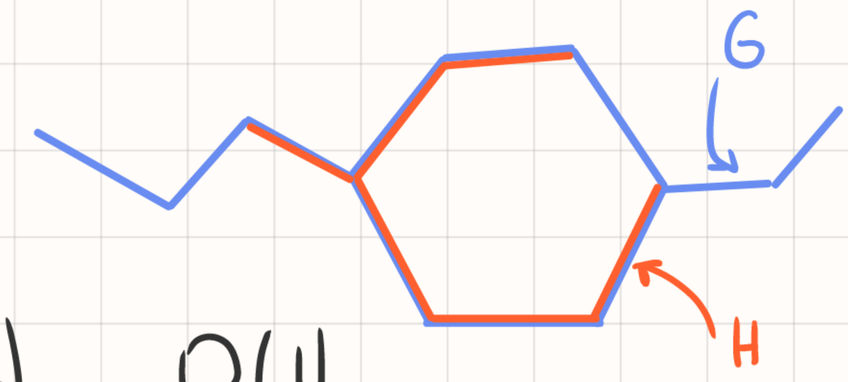
Color Coding

Задача

Дано: паттерн (граф) H на k вершинах и граф G на n вершинах.

Найти: подграф в G изоморфный H



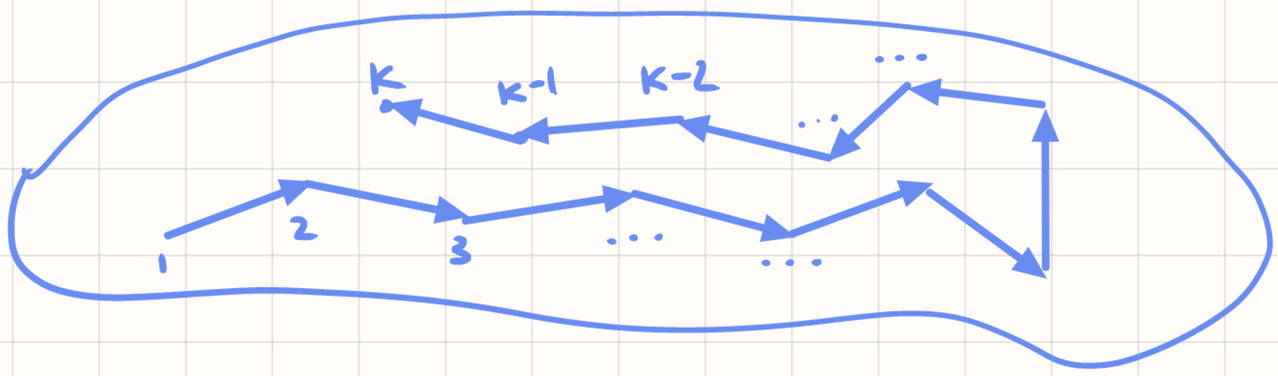
Наивный алгоритм: $O(n^k)$

Если H – лес: FPT решение за $2^{O(k)} n^{O(1)}$

или у H фиксированная
древесная ширина (treewidth)

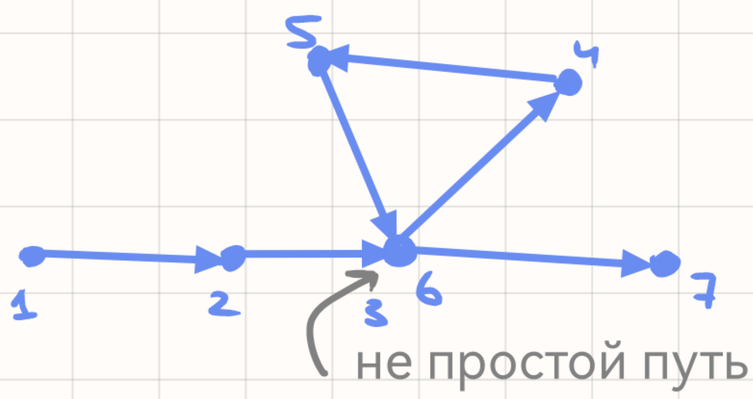
Обсудим простейший случай, когд H – простой путь на k вершинах

Задача Longest path



Дано: граф G и число k.

Проверить: существует ли в G простой путь на k вершинах



▽ сложность состоит в том, что надо найти ПРОСТОЙ путь

Алгоритм Наивное решение

Будем поддерживать уже посещённые вершины и текущую вершину. Тогда сможем решить задачу за $\binom{n}{k} k$

Алгоритм Color coding

1) Раскрасим вершины в k цветов равномено и независимо

$\chi : V \to [k]$

2) Будем искать путь, в котором все вершины различных цветов

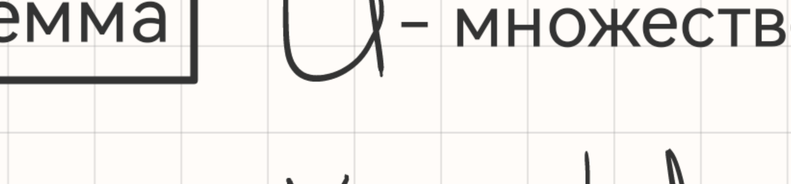
dp[S][u] = [существует ли путь в G, который проходит по цветам из множества S и завершается в вершине u]

База динамики: $|S| = 1$

$$\forall c \in [k] \quad \forall u \in V \quad dp[\{c\}][u] = [c = \chi(u)]$$

Переход:

$$dp[S][u] = \begin{cases} \bigvee\limits_{(v,u)\in E} dp[S \setminus \chi(u)][v] & \chi(u) \in S \\ \text{false} & \text{иначе} \end{cases}$$



3) повторим алгоритм "много" раз, чтобы получить константную вероятность успеха

Далее поймём, сколько раз надо повторить алгоритм и оценим время работы. Для этого рассмотрим следующую лемму:

Лемма U - множество $|U| = n$

$X \subseteq U \quad |X| = k$

$\chi : U \to [k]$ - случайная равномерная независимая раскраска U в k цветов

$$\Rightarrow p = Pr[|\chi(X)| = k] \geqslant e^{-k}$$

↖ $\chi(X) = \{\chi(y) \mid y \in X\}$

□ k^n - количество возможных раскрасок U в k цветов

$k! \, k^{n-k}$ - количество раскрасок, в которых X - разноцветоное множество

$k!$ — S; k^{n-k} — U\S

$$k! > \left(\frac{k}{e}\right)^k \Rightarrow p = \frac{k! \, k^{n-k}}{k^n} = \frac{k!}{k^k} > \frac{k^k}{e^k k^k} = e^{-k}$$

■

Теперь мы готовы оценить время работы алгоритма

Теорема

Существует вероятностный алгоритм, который находит путь размера k в графе G за время $(2e)^k n^{O(1)}$ или сообщает, что найти путь не удалось.

Более того, если алгоритму на вход подать yes-instance (G, k), то он вернет решение с некоторой константной вероятностью

- [ ] Алгоритм, представленный выше работает за $2^k n^{O(1)}$ при фиксированной раскраске

Также мы показали, что вероятность угадать расскраску $\geq e^{-k}$

Тогда повторим алгоритм e^k раз и получим требуемую оценку

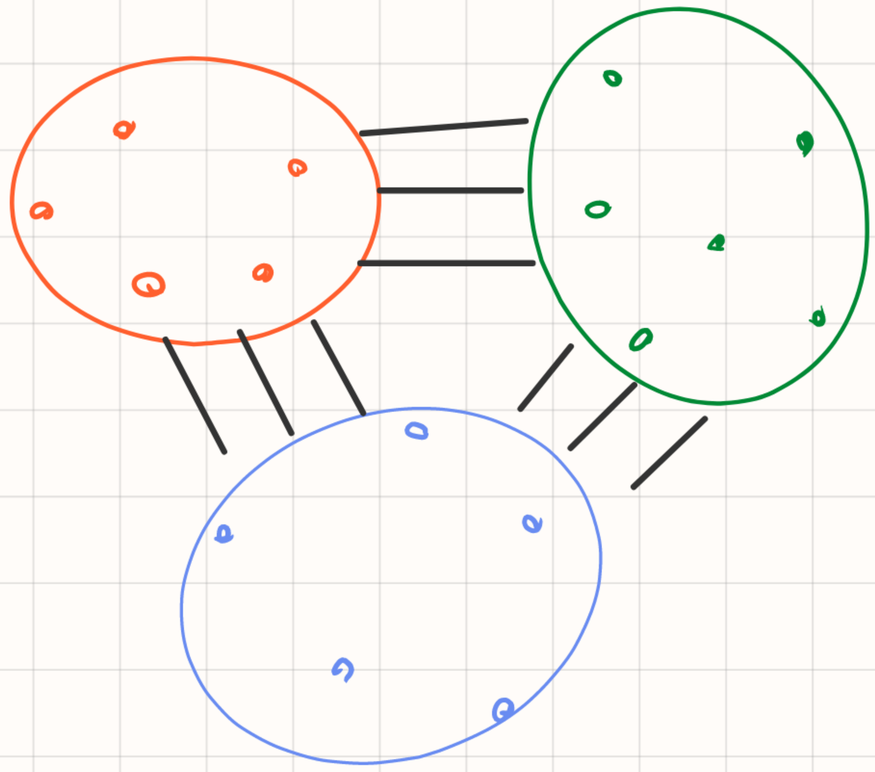
■

A chromatic coding algorithm for d-Clustering

Теперь обсудим задачи, в которых надо модифицировать рёбра.

Идея chromatic coding (также известного как color and conquer) состоит в том, чтобы как и в color coding раскрасить вершины графа и надеяться, что все рёбра решения будут разноцветными.

Когда расскраска зафиксирована, можем пользоваться удобным свойством, а именно: рёбра ответа идут только между классами



Перед тем как углубиться в конкретные примеры, давайте обсудим общие факты

Опр. $\chi: V(G) \to [q]$, назовём граф G правильно раскрашеным если $\forall (u,v) \in E(G) \quad \chi(u) \neq \chi(v)$

Лемма Если в графе G на k рёбрах раскрасить вершины в $\lceil \sqrt{8k} \rceil = O(\sqrt{k})$ цветов равномерно и независимо, то вероятность, что G правильно раскрашен $\geq 2^{-\sqrt{k/2}} = 2^{-O(\sqrt{k})}$

□ $n := |V(G)|$

1). $G_0 = G$

2). $\forall\, i = 1, 2, \dots, n$ выполним следующую процедуру:

- $V_i :=$ вершина минимальной степени графа G_{i-1}

- $G_i := G_{i-1} \setminus V_i$ $\quad d_i = \deg V_i$ в графе G_{i-1}

▽ $G_n = \emptyset$

$$2k = 2|E(G)| \geq 2|E(G_{i-1})| = \sum_{v \in V(G_{i-1})} \deg(v) \geq$$

↖ $\forall_j\ E(G_j) \subseteq E(G)$

$$\geq d_i \cdot |V(G_{i-1})| \geq d_i^2 \implies$$

↖ $d_i \leq |V(G_{i-1})| - 1$

$$\implies d_i \overset{\circledast}{\leq} \sqrt{2k}$$

Теперь рассмотрим процесс раскраски вершин в $q := \lceil \sqrt{8k} \rceil$ цветов в порядке $V_n, V_{n-1}, \dots, V_1$

Обозначим событие $P_i := [G_i$ - правильно раскрашен$]$

▽ $\Pr[P_n] = 1$, $P_j \subseteq P_i \quad \forall j \leq i$

Рассмотрим, что происходит при раскраске вершины V_i :

у вершины есть d_i соседей в графе G_i, поэтому есть $\geq q - d_i$ хороших способов покрасить V_i

$$\implies \Pr[P_{i-1} \mid P_i] \geq \frac{q - d_i}{q} = 1 - \frac{d_i}{q} \overset{(1)}{\geq} 2^{-\frac{2d_i}{q}}$$

(1) : 1). $1 - x \geq 2^{-2x} \quad \forall x \in [0, 1/2]$

2). $d_i \leq \sqrt{2k}$, т.к. $\circledast \implies \frac{d_i}{q} \leq \frac{\sqrt{2k}}{\sqrt{8k}} = \frac{1}{2}$

Тогда получаем, что

$$Pr[P_0] = Pr[P_0 | P_1] Pr[P_1] + Pr[\underbrace{P_0 | \overline{P_1}}_{0,\ т.к.\ P_0 \subseteq P_1}] Pr[\overline{P_1}] =$$

$$= Pr[P_0 | P_1] Pr[P_1] =$$

$$= Pr[P_0 | P_1] Pr[P_1 | P_2] Pr[P_2] =$$

$$= \ldots\ldots =$$

$$= \left(\prod_{i=1}^{h} Pr[P_{i-1} | P_i] \right) \cdot \underbrace{Pr[P_h]}_{1} \geqslant$$

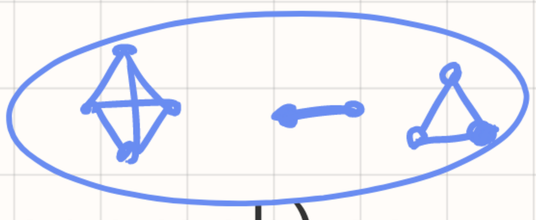
$$\geqslant \prod_{i=1}^{h} 2^{-\frac{2d_i}{9}} = 2^{-\frac{2}{9}\sum d_i} = 2^{-\frac{2k}{\lceil \sqrt{8k} \rceil}} \geqslant 2^{-\sqrt{\frac{k}{2}}}$$

($\sum d_i = k$)

■

Теперь мы готовы рассмотреть данную идею на примере d-clustering problem

Опр. граф H называется ℓ-кластерным графом (ℓ-cluster graph) если в H содержится ℓ компонент связности и каждая из компонент связности является кликой



Опр. H называется кластерным графом (cluster graph) если он является ℓ-кластерным для некоторого ℓ

Опр. Введём операцию $\oplus : Graph \times \binom{V(G)}{2} \to Graph$

$$V(G \oplus A) = V(G) \qquad E(G \oplus A) = (E(G) \setminus A) \cup (A \setminus E(G))$$

Задача d-clustering problem

Дано: граф G и число k

Проверить: существует ли такое множество A, что $G \oplus A$ является d-кластерным графом $\cup$ $|A| \leq k$

▽ d - константа из условия задачи, не часть входа

Опр. множество A из условия задачи называется решением для (G, k)

Алгоритм

① $q := \lceil \sqrt{8k} \rceil$

$\chi : V(G) \to [q]$ - равномерная независимая раскраска

② По лемме решим раскрашенную версию "много" раз и получим вероятностный алгоритм (далее решаем раскрашенную версию)

Опр. $A \subseteq \binom{V(G)}{2}$ называется правильно раскрашенным, если граф (V(G), A) правильно раскрашен

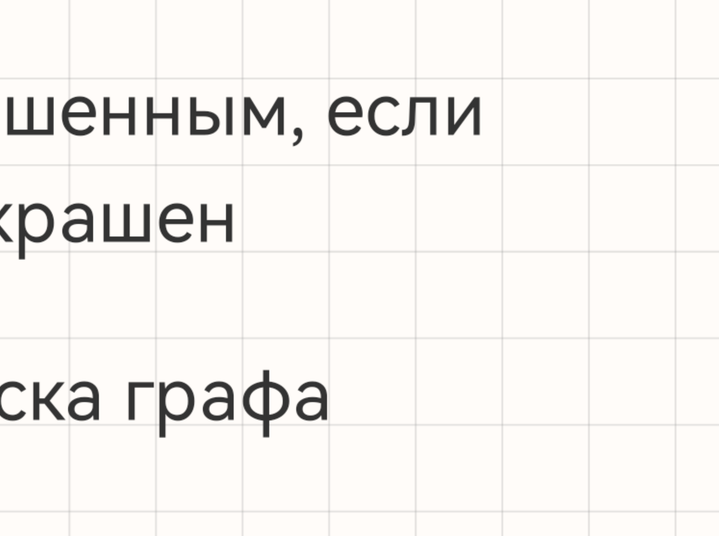
Лемма G - граф, $\chi : V(G) \to [q]$ - раскраска графа

$V_i := \{ v \in V(G) \mid \chi(v) = i \}$

$\Rightarrow$ Если существует решение правильно раскрашенное решение A в G, то $\forall i$ $G[V_i]$ является ℓ-кластерным, где $\ell \leq d$

□ $E(G[V_i]) \cap A \overset{\text{так как все рёбра A - разноцветные}}{=} \varnothing \Rightarrow$

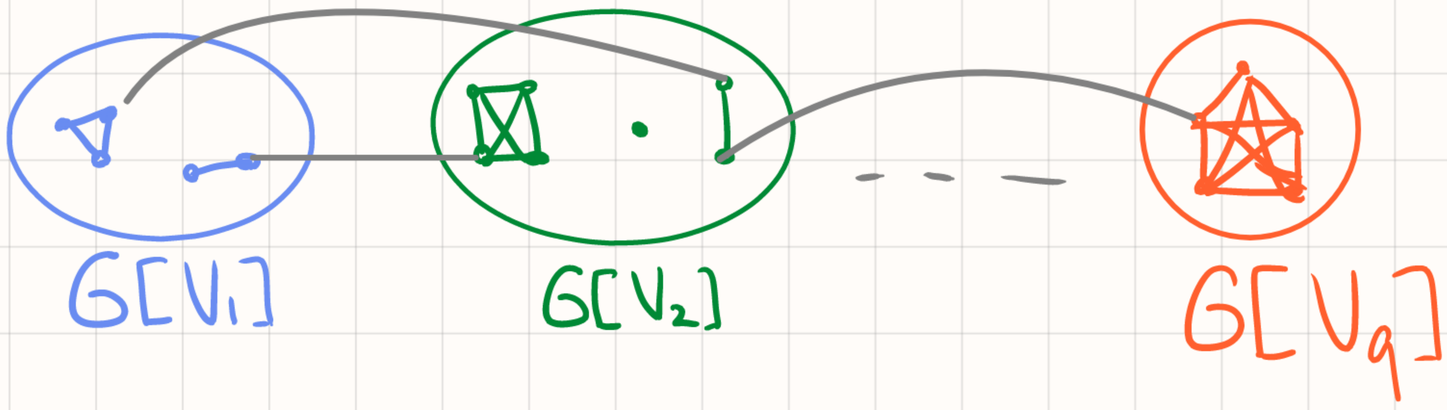
$\Rightarrow G[V_i]$ - подграф графа $G \oplus A$



Но $G \oplus A$ - d-кластерный граф, а любой его подграф является ℓ-кластерным для некоторого $\ell \leq d$

■

Следствие Если для графа G существует решение A, то граф G состоит из множества одноцветных клик (но между разными цветами могли добавить рёбра)



▽ В каждом $G[V_i]$ не более d клик. Поэтому "клик" в графе G не более чем $dq = d\lceil\sqrt{8k}\rceil$

③ (решаем раскрашенную версию задачи)

Проверим, что граф разбивается на одноцветные "клики". Если не разбивается, то ответ для данной раскраски – нет

Далее решим "тупым" перебором. Для кадой "клики" решим, к какому из d кластеров она будет принадлежать. Когда d кластеров фиксированы, понятно, какие рёбра нужно добавить или удалить. Остаётся проверить, что количество добавленных и удалённых рёбер не превосходит k.

Таким образом, надо перебрать не более $d^{d\lceil\sqrt{8k}\rceil} = 2^{O(d \log d \sqrt{k})}$ вариантов

И итоговое время работы 3-го пункта: $2^{O(d \log d \sqrt{k})} n^{O(1)}$

③' на самом деле можно выполнить 3-й пункт быстрее.

$dp[C][f][S][t]$ = [можно ли получить t-кластерный граф, постренный на "кликах" из множества C, если в множестве A f рёбер, а последний кластер – это множество S]

Обозначим "клики": $C_1, C_2, \dots, C_{dq}$

База динамики: $|C| = 0$

$$dp[\emptyset][f][S][t] = \begin{cases} true & f = t = 0, S = \emptyset \\ false & \text{otherwise} \end{cases}$$

Переход динамики:

1 сл. Сделать очередную "клику" новым последним кластеорм

$$dp[C \sqcup \{c_i\}][f'][\{c_i\}][t+1] \mid = dp[C][f][S][t]$$

формульно f' выписывать не будем, но понятно, что можно пробежаться по графу и пересчитать количество добавленных/удалёных рёбер

2 сл. Добавить очередную "клику" к последнему кластеру

$$dp[\underbrace{C \sqcup \{c_i\}}_{2^{dq}}][\underbrace{f'}_{K}][\underbrace{S \cup \{c_i\}}_{2^{dq}}][\underbrace{t}_{d}] \mid = dp[C][f][S][t]$$

Тогда время работы для раскрашенной версии:

$$3^{dq} \cdot k \cdot d \cdot n^{O(1)} = 2^{O(d\sqrt{k})} \cdot n^{O(1)}$$

Теорема

Существует вероятностный алгоритм, который для (G, k)-instance задачи d-clustering за время $2^{O(d\sqrt{k})} n^{O(1)}$ либо сообщает о неудаче, либо возвращает решение для (G, k). Более того, если алгоритму дать yes-instance, то он вернёт решение с константной вероятностью

□ Выше показали, что вероятность хорошей раскраски $\geq \frac{1}{2^{O(\sqrt{k})}}$

Тогда повторим описанный выше алгоритм $2^{O(\sqrt{k})}$ раз и получим нужную оценку ■