ЛЕКЦИЯ 6

ХРОМАТИЧЕСКОЕ ЧИСЛО. ЖИРНЫЙ АЛГОРИТМ. ОБХВАТ ГРАФА

1. Хроматическое число случайного графа

Вспомним упражнение 7, которое было сформулировано в конце предыдущей лекции. Оно состояло из трех задач, первые две из которых были очень простыми. Оказывается, есть еще несколько более сложных случаев для немного большей вероятности:

Теорема 20 (Боллобаш) Пусть:

$$p = n^{-\alpha}, \quad \alpha > \frac{2}{3}.$$

Тогда существует функция u:

$$\exists u = u(n,p): \text{ асимптотически почти наверное } u \leqslant \chi(G) \leqslant u+1.$$

Теорема 21 Пусть $p = \frac{1}{2}$ Тогда асимптотически почти наверное:

$$\alpha(G) \leqslant 2\log_2 n, \quad \omega(G) \leqslant 2\log_2 n,$$

следовательно:

$$\chi(G) \geqslant \frac{n}{2\log_2 n}.$$

Теорема 22 (Боллобаш) Существует такая функция ϕ что:

$$\phi(n) = o\Big(\frac{n}{\ln n}\Big): \quad P\left(\Big|\chi(G) - \frac{n}{2\log_2 n}\Big| \leqslant \phi(n)\right) \to 1, \quad n \to \infty.$$

Задача Пояснить, почему при некоторых p из диапазона:

$$p = n^{-\alpha}, \quad \alpha > \frac{2}{3}$$

функция u с ростом n заведомо стремится к бесконечности. $*$

2. Жирный алгоритм

Задача отыскания максимальной клики в графе, задача отыскания максимального независимого множества, задача о правильной раскраске (то есть задачи о нахождении $\omega(G), \alpha(G), \chi(G)$) — являются **трудными с алгоритмической точки зрения**. Это элемент теории сложности вычислений.

Другими словами, не существует алгоритмов, имеющих полиномиальную сложность (время работы алгоритма есть полином от числа вершин или ребер графа) и решающих эту задачу на произвольном графе на n вершинах.

Тогда возникает задача приближенного нахождения этих характеристик с помощью алгоритмов с полиномиальной сложностью. Продемонстрируем, как с помощью теории случайных графов можно оценивать качество работы различных алгоритмов, что имеет почти практический смысл.

Рассмотрим понятие **жадный алгоритм раскраски**. Имеется граф, произвольным образом занумеруем его вершины (см. рис. 6.1).

Первую вершину красим в первый цвет. Смотрим на вторую вершину. Если между первой и второй вершинами есть ребро, то красим вторую вершину во второй цвет, иначе — в первый. Далее берем третью вершину и стараемся покрасить её в цвет с наименьшим номером, в который её только можно покрасить. Продолжим раскраску для оставшихся вершин графа.

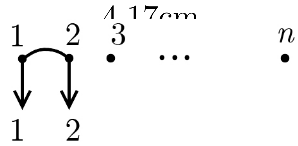

Рис. 6.1

Будем рассматривать этот алгоритм подробнее. Покажем, что в каком-то смысле этот алгоритм очень хорош, что не является очевидным фактом.

Немного упростим задачу и применим этот аппроксимационный алгоритм не для раскраски, а для отыскания максимального независимого множества вершин. Тогда пусть $\alpha'(G)$ и $\chi'(G)$ — характеристики графа, найденные с помощью жадного алгоритма.

Посчитаем, в какой из цветов будет покрашено больше всего вершин, и это будет аппроксимацией для числа независимости графа, так как множества из вершин одного цвета очевидно являются независимыми.

Теорема 23 Если $p = \frac{1}{2}$, то асимптотически почти наверное:

$$\alpha'(G) \geqslant (1-\epsilon) \cdot \log_2 n,$$

где ϵ — любая наперед заданная сколь угодно малая константа. *

Док-во: Попробуем оценить сверху:

$$P(A), \text{ где событие } A = \alpha'(G) < (1-\epsilon) \cdot \log_2 n.$$

Отметим, что утверждения «A — подмножество B» и «A влечет B ($A \Rightarrow B$)» — это одно и то же.

Обозначим:

$$M = \left[\frac{n}{2(1-\epsilon)\log_2 n}\right].$$

Пользуясь рис. 6.2, проведем следующие рассуждения. В нашем случае видно, что жадный алгоритм нашел много маленьких множеств, объединяющих элементы одного цвета ($C_1, \ldots C_m$). Почему же в данном графе жадный алгоритм не смог увеличить получившиеся островки?

Пусть x — вершина, не принадлежащая ни одному из островков. Знаем, что жадный алгоритм устроен так, что из x идет хотя бы одно ребро в каждый из островков, иначе x был бы присоединен к тому или иному островку.

Получим, что любая вершина из дополнения имеет хотя бы одно ребро, ведущее в каждый островок, что можно сформулировать следующим образом:

$$A \Rightarrow \quad \exists\, a_1, \ldots, a_m: \ a_i < (1-\epsilon)\log_2 n, \\ \exists\, C_1, \ldots, C_m: \ C_i \subset V, |C_i| = a_i, \quad C_i \cap C_j = \emptyset, \\ \forall x \in V \ \ (C_1 \sqcup \ldots \sqcup C_m), \ \forall i \quad \exists y \in C_i: \ x \sim y.$$

Утверждаем, что:

$$P(A) \leqslant \sum_{a_1=1}^{(1-\epsilon)\log_2 n} \ldots \sum_{a_m=1}^{(1-\epsilon)\log_2 n} \sum_{C_1,\ldots,C_m \subset V:\ |C_1|=a_1,\ldots,|C_m|=a_m,\ C_i \cap C_j = \emptyset} \Big(P(\forall x \in V \ \ (C_1 \sqcup \ldots \sqcup C_m), \forall i \quad \exists y \in C_i: \ x \sim y)\Big) \tag{6.1}$$

Найдем в точности, чему равна вероятность, стоящая под суммами в предыдущем выражении. Пусть:

$$x \in V \ \ (C_1 \sqcup \ldots \sqcup C_m) \text{ — фиксированная вершина.}$$

Тогда вероятность того, что из x ни одно ребро не ведет в C_i есть $\left(\frac{1}{2}\right)^{a_i}$. Значит:

$$P(\exists y \in C_i: \ x \sim y) = 1 - \left(\frac{1}{2}\right)^{a_i},$$

$$P(\forall i\ \exists y \in C_i: \ x \sim y) = \prod_{i=1}^{m} 1 - \left(\frac{1}{2}\right)^{a_i}.$$

Для разных x события будут независимы, следовательно (здесь не будем писать пределы для сумм из выражения (6.1), подразумевая, что они остались теми же):

$$\sum \dots \sum \sum \Big(P(\forall x \in V \ \ (C_1 \sqcup \dots \sqcup C_m), \forall i \quad \exists y \in C_i : \ x \sim y) \Big) =$$

$$= \sum \dots \sum \sum \left(\prod_{i=1}^{m} \left(1 - \left(\frac{1}{2}\right)^{a_i} \right) \right)^{n - \sum_{i=1}^{m} a_i} \leqslant$$

$$\leqslant \sum \dots \sum \sum \left(1 - \left(\frac{1}{2}\right)^{(1-\epsilon)\log_2 n} \right)^{m\left(n - \sum_{i=1}^{m} a_i\right)} \leqslant$$

$$\leqslant \sum \dots \sum \sum \left(1 - \left(\frac{1}{2}\right)^{(1-\epsilon)\log_2 n} \right)^{m - \frac{n}{2}} <$$

$$< (\log_2 n)^m \cdot (C_n^{log_2 n})^m \cdot \left(1 - \frac{1}{n^{1-\epsilon}}\right)^{\frac{nm}{2}} < (\log_2 n)^m \cdot n^{m \log_2 n} \cdot \exp\left(-\frac{nm}{2n^{1-\epsilon}}\right).$$

Теперь проделаем следующие преобразования:

$$\exp\left(-\frac{nm}{2n^{1-\epsilon}}\right) = \exp\left(-\frac{n^{\epsilon} m}{2}\right) = \exp\left(\frac{n^{\epsilon}}{2} \cdot \frac{n}{2(1-\epsilon)\log_2 n}(1 + o(1))\right) = \exp\Big(-n^{1+\delta}\Big).$$

Здесь было замечено, что логарифм растет медленнее любой положительной степени n, каким бы малым ϵ ни был.

Тогда убедимся в том, что и остальные множители стремятся к нулю, чем завершим доказательство:

$$(\log_2 n)^m \cdot n^{m \log_2 n} \cdot \exp\left(-\frac{nm}{2n^{1-\epsilon}}\right) < (\log_2 n)^m \cdot n^{m \log_2 n} \cdot \exp\Big(-n^{1+\delta}\Big), \quad n > n_0.$$

$$n^{m \log_2 n} = \mathrm{e}^{m(\log_2 n)\ln n} = \mathrm{e}^{\mathrm{const}\cdot m \log_2^2 n} = \exp\left(\mathrm{const}(1 + o(1)) \cdot \frac{n \log_2^2 n}{2(1-\epsilon)\log_2 n}\right) =$$

$$= \mathrm{e}^{c'\big(1+o(1)\big) n \log_2 n} \to 0,$$

что доводит до конца доказательство теоремы. ■

Таким образом, доказано, что почти на всех графах жирный алгоритм ошибается не более, чем в два раза. Но бывают графы, на которых он ошибается почти в n раз. Сформулируем следующую теорему о таких графах:

Теорема 24 (Кучеры)

$$\forall \epsilon > 0, \ \delta > 0 \ \exists \text{ последовательность } G_n : \ |V(G_n)| = n, \ P_\sigma\left(\frac{\alpha(G_n)}{\alpha'_\sigma(G_n)} \geqslant n^{1-\epsilon}\right) > 1 - \delta,$$

где σ — нумерация вершин $V(G_n)$,

$\alpha'_\sigma(G_n)$ — оценка числа независимости, найденная жирным алгоритмом в нумерации σ,

P_σ — вероятность, вычисленная в предположении, что нумерация выбрана случайно, то есть с вероятностью $\frac{1}{n!}$. $*$

Итак, видно, что жирный алгоритм не стоит использовать, если нельзя ошибиться сильно, ведь, к примеру, результат «ядерный реактор почти всегда не взорвется» будет неудовлетворительным.

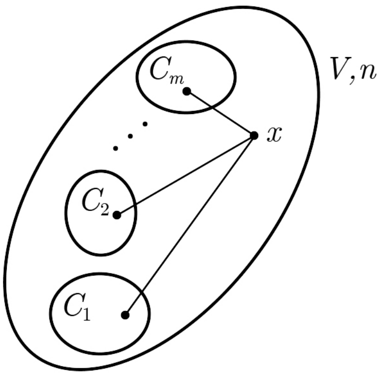

Рис. 6.2

3. Обхват графа

Определение 15: *Обхватом графа G называется длина кратчайшего цикла в этом графе. Обозначается $g(G)$ от английского слова girth (обхват).* ♣

Рассмотрим теорему, которая продемонстрирует одно из самых красивых применений теории случайных графов:

Теорема 25 (Эрдеш, 1959)

$$\forall l, k \ \exists G: \quad \chi(G) > k, \quad g(G) > l.$$

Док-во: Сформулируем идею доказательства, а полную техническую реализацию осуществим на следующей лекции.

Возьмем $\theta = \frac{1}{2l}$, тогда вероятность ребра случайного графа положим:

$$p = n^{\theta - 1} \in (0, 1).$$

Рассмотрим случайный граф $G(n, p)$. Пусть случайная величина $X(G)$ — это количество циклов в G, имеющих длину не больше l, то есть количество циклов, от которых хотелось бы избавиться. Тогда:

$$MX = \sum_{i=1}^{l} C_n^i \cdot \frac{(i-1)!}{2} \cdot p^i < \sum_{i=1}^{l} (np)^i = \sum_{i=1}^{l} n^{\theta i} < l \cdot n^{\theta l} = l \cdot n^{\frac{1}{2}}.$$

Значит, по неравенству Маркова:

$$P\Big(X > \frac{n}{2}\Big) \leqslant \frac{MX}{\frac{n}{2}} \to 0,$$

следовательно:

$$P\Big(X \leqslant \frac{n}{2}\Big) \to 1.$$

Тогда если удастся найти граф, у которого не только $X \leqslant \frac{n}{2}$, но к тому же и $\alpha(G)$ маленькое, то удаление из него вершин не будет приводить к уменьшению $\alpha(G)$, а $\chi(G)$ оценим количеством вершин (которое не меньше, чем $\frac{n}{2}$) поделенным на данное $\alpha(G)$.

С ростом n хроматическое число окажется большим, чем k, а циклов не будет, так как мы их разорвали.

Реализацией сформулированной выше идеи займемся на следующей лекции. ■