Лекция 4

Приближенные алгоритмы для задач о максимальном сечении и о мощности объединения множеств

(Конспект: М. Плискин)

4.1 Приближенные алгоритмы

Рассмотрим некоторое множество S и функцию $f\colon S \to \mathbb{R}_+$. Предположим, что существует элемент $x_{\text{опт}} \in S$, на котором функция f достигает своего максимального значения. Наша задача состоит в том, чтобы найти этот элемент $x_{\text{опт}}$.

Мы будем рассматривать случаи, когда эффективного алгоритма точного решения нашей задачи не существует или он неизвестен. Поэтому мы будем искать приближенное решение нашей задачи в следующем смысле:

Определение 4.1. Алгоритм, с помощью которого можно найти такой элемент $x' \in S$, что

$$f(x') \geq \alpha f(x_{\text{опт}}),$$

называется *α-приближенным алгоритмом* решения данной задачи. □

Данное определение, очевидно, корректно лишь в случае детерминированных алгоритмов. Для недетерминированных алгоритмов установим следующее, более общее, определение:

Определение 4.2. Вероятностный алгоритм, в результате работы которого генерируется элемент $x_{\text{ответ}} \in S$ и при этом

$$\mathbf{E}\, f(x_{\text{ответ}}) \geq \alpha f(x_{\text{опт}}),$$

называется *α-приближенным вероятностным алгоритмом* решения данной задачи. □

4.2 Приближенный алгоритм для задачи о максимальном сечении

Определение 4.3. Задача о максимальном сечении (MAX-CUT). Пусть есть граф $G = (V, E)$, каждому ребру $e \in E$ которого приписан некоторый вес w_e. Рассмотрим все возможные раскраски вершин графа в белый и черный цвета (то есть отображения $j\colon V \to \{\text{Ч}, \text{Б}\}$), и каждой такой раскраске M сопоставим число $f(M)$, вычисляемое как сумма весов ребер, концы которых имеют в рассматриваемой раскраске M разные цвета (множество таких ребер назовем *сечением*). Наша задача состоит в том, чтобы найти такую раскраску M_0, чтобы $f(M_0)$ было бы максимальным. □

Будем решать эту задачу приближенно. Сначала переформулируем задачу как задачу оптимизации:

Определение 4.4. MAX-CUT как задача целочисленного линейного программирования. Сопоставим каждой вершине $v_i \in V$ с номером i число y_i, вычисляемое как

$$y_i = \begin{cases} -1, & v_i \text{ — белая вершина} \\ 1, & v_i \text{ – черная вершина.} \end{cases}$$

Тогда ребро $\{v_i, v_j\}$ принадлежит сечению в том, и только в том случае, когда

$$y_i y_j = -1,$$

то есть

$$f(M) = f((y_1, y_2, \dots, y_n)) = \sum_{i<j} \frac{1 - y_i y_j}{2} w_{ij}.$$

Наша задача состоит в том, чтобы найти максимум $f((y_i)_i)$ по всем наборам $(y_i)_i$, где каждое $y_i \in \{-1, 1\}$. □

Обозначим решение нашей задачи через M^* и попытаемся сначала решить более простую задачу:

Определение 4.5. Релаксация MAX-CUT: оптимизация на сфере большей размерности. Будем искать

$$\max_{v_1,\dots,v_n \in S^n} g((v_1,\dots,v_n)),$$

где

$$g((v_1,\dots,v_n)) = \sum_{i>j} \frac{1 - v_i \cdot v_j}{2} w_{ij},$$

а S^n — n-мерная единичная сфера (n — число вершин в графе G); умножение понимается как скалярное произведение. □

Обозначим этот максимум через R^*. Отметим, что

$$M^* \le R^*.$$

Этот факт следует из того, что наша новая задача содержит старую как частный случай (достаточно зафиксировать все переменные, кроме одной).

Теперь нам необходимо описать способ получения решения старой задачи из решения новой.

Алгоритм 4.1 (Randomized rounding). Пусть у нас есть набор $(v_i)_i$, являющийся решением новой задачи. Можно считать, что начало у всех векторов помещено в начало координат. Проведем через начало координат случайным образом гиперплоскость T размерности $n-1$. Эта гиперплоскость разделит пространство на два полупространства A и B. Теперь по каждому вектору v_i построим число y_i следующим образом:

$$y_i = \begin{cases} 1, & v_i \in A \\ -1, & v_i \in B. \end{cases}$$

Иначе говоря, выберем случайным образом вектор $u \in S^n$ и построим число y_i так:

$$y_i = \begin{cases} 1, & u \cdot v_i > 0 \\ -1, & u \cdot v_i < 0 \end{cases}$$

(случай $u \cdot v_i = 0$ будем интерпретировать как попало: вероятность этого события равна нулю). □

Теперь докажем следующую лемму про только что описанный алгоритм.

Лемма 4.1. *Пусть у нас теперь есть набор векторов v_i — (приближенное) решение задачи 4.5, R' — значение g на этом наборе. Пусть также $(y_i)_i$ - набор, полученный с помощью вышеописанного алгоритма из набора векторов v_i. Обозначим также $U' = f((y_i)_i)$. Тогда*

$$\mathbf{E}\, U' \geq \alpha R',$$

где $\alpha \approx 0.87$.

Доказательство. Нам надо доказать следующее неравенство:

$$\mathbf{E} \sum_{i<j} \frac{1 - y_i y_j}{2} w_{ij} \geq \alpha \sum_{i<j} \frac{1 - v_i \cdot v_j}{2} w_{ij}.$$

Возьмем некоторые y_i и y_j. Тогда $y_i y_j = -1$ в том, и только в том случае, когда вектора v_i и v_j лежат в разных полупространствах относительно нашей гиперплоскости T. Вычислим вероятность того, что $y_i \neq y_j$:

$$P(y_i \neq y_j) = \frac{\theta_{ij}}{\pi},$$

где θ_{ij} — угол между векторами v_i и v_j. Отсюда получаем, используя определение математического ожидания:

$$\mathbf{E} \sum_{i<j} \frac{1 - y_i y_j}{2} w_{ij} = \sum \frac{\theta_{ij}}{\pi} w_{ij}.$$

При этом

$$\sum_{i<j} \frac{1 - v_i \cdot v_j}{2} w_{ij} = \sum_{i<j} \frac{1 - \cos\theta_{ij}}{2} w_{ij}.$$

Теперь осталось взять

$$\alpha = \min_{\theta} \frac{2}{\pi} \frac{\theta}{1 - \cos\theta} \approx 0.87,$$

и утверждение леммы доказано. □

Определение 4.6. Матрица Y является положительно полуопределенной ($Y \succcurlyeq 0$), если для любого вектора x верно $x^{\mathrm{T}} Y x \geq 0$. □

Теперь еще раз переформулируем нашу задачу.

Определение 4.7. MAX-CUT как задача полуопределенного программирования. Необходимо максимизировать выражение

$$\sum \frac{1-y_{ij}}{2} w_{ij}$$

по всем матрицам

$$Y = \{y_{ij}\}_{i,j=1}^{n}$$

таким что $Y \succcurlyeq 0$ и $\forall i \; y_{ii} = 1$. $\square$

Приведем следующие три факта без доказательств:

Факт 4.1. *Задачу 4.7 можно решить приближенно за полиномиальное время и при этом получить результат $R^* - \varepsilon$.*

Факт 4.2. *Если матрица Y положительно полуопределена ($Y \succcurlyeq 0$), то она является матрицей скалярных произведений некоторого набора векторов v_i (заметим, что условие $y_{ii} = 1$ гарантирует, что эти вектора — единичные), причем эти вектора можно найти за полиномиальное время.*

Теорема 4.1. *Мы построили полиномиальный по времени вероятностный $(\alpha - \varepsilon)$-приближенный алгоритм для $\alpha \approx 0.87$ и произвольного $\varepsilon > 0$.*

Доказательство. Решение нашей задачи, согласно факту 4.2, свелось к решению задачи 4.7. Она, по факту 4.1, решается за полиномиальное время с аддитивной погрешностью не более ε'. Таким образом, по лемме, мы построили искомый $(\alpha - \varepsilon)$-приближенный алгоритм. $\square$

Факт 4.3. *Этот алгоритм можно сделать детерминированным.*

Факт 4.4. *Если бы существовал такой алгоритм для $\alpha > 0.95$, то было бы $\mathbf{P} = \mathbf{NP}$.*

Факт 4.4 утверждает, что мы получили довольно хороший результат с теоретической точки зрения.

4.3 Метод Монте-Карло

Рассмотрим следующую проблему, связанную с теорией вероятностей: пусть мы проводим m испытаний, вероятность успеха в каждом из которых есть p_i. Определим случайную величину y_i следующим образом:

y_i принимает значение 1 с вероятностью p_i и значение 0 с вероятностью $1 - p_i$. Определим

$$\mu = \mathbf{E} \sum_{i=1}^{m} y_i = \sum_{i=1}^{m} p_i.$$

Теперь, если обозначить

$$Y = \sum_{i=1}^{m} y_i,$$

то справедливы следующие *оценки Чернова*:

Факт 4.5.

$$P\{Y > (1+\varepsilon)\mu\} < \left(\frac{e^{\varepsilon}}{(1+\varepsilon)^{1+\varepsilon}}\right)^{\mu} \leq e^{-\frac{\mu\varepsilon^2}{4}},$$

$$P\{Y < (1-\varepsilon)\mu\} < e^{-\frac{\mu\varepsilon^2}{2}}.$$

Теперь рассмотрим собственно метод Монте-Карло.

Определение 4.8. Задача о нахождении мощности подмножества. Пусть у нас есть два конечных множества: U и его подмножество G. При этом мощность множества U известна. Мы хотим найти мощность множества G. □

Для решения этой задачи существует следующий хорошо известный вероятностный алгоритм, называемый *методом Монте-Карло.*

Алгоритм 4.2. Будем брать случайным образом точки множества U и проверять их на принадлежность к множеству G. Произведя m экспериментов, определим величину

$$\widetilde{\rho} = \frac{\sum_{i=1}^{m} y_i}{m}$$

и будем считать, что оня является приближенным значением величины

$$\rho = \frac{|G|}{|U|}.$$

Отсюда найдем приближенно мощность множества G. □

Очевидно, что чем больше экспериментов мы произведем, тем точнее мы определим ρ. Из оценок Чернова вытекает следующее соотношение:

$$P\{\widetilde{\rho} \notin [(1-\varepsilon)\rho; (1+\varepsilon)\rho]\} \leq 2e^{-\frac{\rho\varepsilon^2}{4}},$$

из которого следует основная теорема:

Теорема 4.2. *Для того, чтобы с вероятностью как минимум* $1-\delta$ *получить* $\widetilde{\rho}$, *удовлетворяющее*

$$(1-\varepsilon)\rho \le \widetilde{\rho} \le (1+\varepsilon)\rho,$$

достаточно провести

$$\frac{4}{\varepsilon^2\rho}\ln\frac{2}{\delta}$$

экспериментов.

Таким образом, чтобы знать, сколько экспериментов надо произвести, надо знать значение ρ, но это как раз то, что мы ищем! Эта проблема преодолима, если мы знаем какую-нибудь не слишком маленькую нижнюю оценку для ρ.

4.4 Мощность объединения множеств

Рассмотрим следующую задачу:

Определение 4.9. Задача о мощности объединения множеств. Пусть есть n множеств H_i, мощности которых известны. Необходимо найти мощность их объединения $\bigcup_{i=1}^n H_i$. □

Для решения этой задачи воспользуемся методом Монте-Карло. Введем следующие обозначения:

$$\begin{aligned}
G &= \bigcup_{i=1}^{n} H_i,\\
U &= \{(v,i) \mid v \in H_i\},\\
G' &= \left\{(v,i) \in U \;\middle|\; i = \min_{k\in 1..n}\{k : v \in H_k\}\right\}.
\end{aligned}$$

Отметим, что мощность множества G' совпадает с мощностью нужного нам объединения:

$$|G'| = \left|\bigcup_{i=1}^{n} H_i\right|.$$

Определим теперь величину ρ как

$$\rho = \frac{|G'|}{|U|}.$$

Эту величину мы можем узнать из метода Монте-Карло. Мощность множества U нам также известна:

$$|U| = \sum_{i=1}^{n} |H_i|.$$

Таким образом, мы найдем мощность множества G', то есть решим нашу задачу.

Осталось лишь оценить количество испытаний. Для этого оценим величину ρ снизу:

$$\rho \geq \frac{1}{n}.$$

Эта оценка вытекает из того, что (нестрогое пояснение) «худший случай» достигается, когда все H_i совпадают, а их мощность равна N. Тогда мощность множества U равна nN, а мощность объединения равна N, откуда и вытекает требуемое соотношение.

Отсюда получаем выражение для количества испытаний

$$m \geq \frac{4}{\varepsilon^2 \rho} \ln \frac{2}{\delta} \geq \frac{4n}{\varepsilon^2} \ln \frac{2}{\delta},$$

достаточного для того, чтобы вычисленное значение ρ отклонялось от истинного не более чем на ε с вероятностью $1 - \delta$.