

МОСКОВСКИЙ ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ

ПОИСК МАКСИМАЛЬНОГО РАЗРЕЗА

Проект по сложности вычислений

Автор:
Седова Анна

Contents

1	\mathcal{NP}-полнота	3
2	Аппроксимационный алгоритмы для решения задачи	4
2.1	Наивный алгоритм	4
2.2	Алгоритм Геманса и Вильямсона. Описание алгоритма	4
2.3	Алгоритм Геманса и Вильямсона. Оценка работы	5
2.4	Сравнение наивного алгоритма и алгоритма Геманса и Вильямсона	6

Chapter 1

\mathcal{NP} -полнота

Задача MAXCUT

Рассмотрим задачу $\text{MAXCUT} = (G, k) \mid$ в неориентированном взвешенном графе G есть разрез размера не меньше k . Покажем, что она является \mathcal{NP} -полной.

$\text{MAXCUT} \in \mathcal{NP}$

Proof. В качестве сертификата рассмотрим номера вершин, входящие в разрез, а верификатор проверит, что размер этого разреза не меньше k . Так как для проверки размера разреза достаточно посчитать сумму весов ребер внутри разреза и сравнить с k , то верификатор будет работать за полиномиальное время. □

MAXCUT является \mathcal{NP} -трудной

Proof. 1) **PARTITION**

$\text{PARTITION} = \{(c_1, c_2, \dots, c_n) \in \mathbb{Z}^n \mid \text{существует разбиение на два множества } S_1, S_2 : \sum_{i \in S_1} c_i = \sum_{j \in S_2} c_j\}$

УТВ(6/д) **PARTITION** является \mathcal{NP} -полной задачей.

2) **Покажем, что $\text{PARTITION} \leq_p \text{MAXCUT}$**

Построим полный граф на n вершинах, пусть вес ребра между i -й и j -й вершинами будет равен $c_i c_j$. Будем искать разрез размера не меньше $k = \frac{1}{4} \sum_{i=1}^n c_i^2$.

Обозначим вершины одной доли при таком разрезе S_1 , а при втором S_2 . Пусть такой разрез существует. Тогда его вес $\leq \sum_{i \in S_1, j \in S_2} c_i c_j = \sum_{i \in S_1} c_i * \sum_{j \in S_2} c_j \leq (\frac{1}{2} \sum_{i=1}^n c_i)^2 = \frac{1}{4} \sum_{i=1}^n c_i^2$, то есть ровно $\frac{1}{4} \sum_{i=1}^n c_i^2$. Тогда в неравенстве $\sum_{i \in S_1} c_i * \sum_{j \in S_2} c_j \leq (\frac{1}{2} \sum_{i=1}^n c_i)^2$ достигается равенство, то есть $\sum_{i \in S_1} c_i = \sum_{j \in S_2} c_j = \frac{1}{2} \sum_{i=1}^n c_i$, то есть разбиение существует.

Пусть такого разреза не существует. Покажем от противного, что и разбиения не существует. Если бы разбиение существовало, то существовал бы разрез мощности $\leq (\frac{1}{2} \sum_{i=1}^n c_i)^2 = \frac{1}{4} \sum_{i=1}^n c_i^2$, то есть разрез искомого размера. Противоречие, следовательно, разбиения не существует.

Так как **PARTITION** является \mathcal{NP} -трудной, то и **MAXCUT** является \mathcal{NP} -трудной. □

Задача поиска

Так как задача является \mathcal{NP} -полной, то задача поиска будет иметь такую же сложность.

Chapter 2

Аппроксимационный алгоритмы для решения задачи

Так как MAXCUT - \mathcal{NP} -полная, то неизвестно, существует ли алгоритм, решающий задачу за полином. Поэтому здесь будет рассматриваться алгоритм, решающий задачу приближенно.

2.1 Наивный алгоритм

Рассмотрим самый простой аппроксимационный алгоритм. Случайно распределим вершины по долям. Оценим $E(\text{мощность разреза}) = \sum_{i=0, j=0}^{n, n} w_{i,j} * P(i \text{ и } j \text{ в разных долях}) = \sum_{i=0, j=0}^{n, n} w_{i,j} * 0.5 \geq 0.5 * \text{мощность максимального разреза}$. Таким образом, наивный алгоритм дает аппроксимационный коэффициент 50

2.2 Алгоритм Геманса и Вильямсона. Описание алгоритма

На входе дается граф $G = (V, E)$ размера n и с матрицей весов W . Хотим найти максимальный разрез с большой точностью.

Сведение к задаче оптимизации

Рассмотрим разрез W , при котором вершины поделены на множества S_1 и S_2 . Пусть $x_i = 1$, если $x_i \in S_1$ и $x_i = -1$, если $x_i \in S_2$ при $i = 1 \dots n$. Тогда мощность этого разреза будет равна $|W| = \frac{1}{8} \sum_{i=0, j=0}^{n, n} w_{i,j} * (x_i - x_j)^2 = \frac{1}{8} \sum_{i=0, j=0}^{n, n} w_{i,j} * (x_i^2 + x_j^2 - 2x_i x_j)$. Заметим, что $x_i^2 = 1$, то есть $|W| = \frac{1}{8} \sum_{i=0, j=0}^{n, n} w_{i,j} * (2 - 2x_i x_j) = \frac{1}{4} \sum_{i=0, j=0}^{n, n} w_{i,j} * (1 - x_i x_j)$ (1)

Таким образом, поиск максимального разреза сводится к следующей задаче оптимизации $\max_{x_i \in \mathbb{Z}} \frac{1}{4} \sum_{i=0, j=0}^{n, n} w_{i,j} * (1 - x_i x_j)$, где $x_i^2 = 1$, что эквивалентно $\min_{x_i \in \mathbb{Z}} \sum_{i=0, j=0}^{n, n} w_{i,j} * (x_i x_j)$ (2), где $x_i^2 = 1$. Хотелось бы приблизить эту задачу оптимизации задачей, которая может быть решена за полиномиальное время. Сделаем это следующим образом.

Рассмотрим вместо x_i векторы $y_i \in R^n$ и запишем аналогичную задачу оптимизации: $\min_{y_i \in R^n} \sum_{i=0, j=0}^{n, n} w_{i,j} * \langle y_i, y_j \rangle$ (3), где $\langle y_i, y_i \rangle = 1$, что эквивалентно $\min_{U \in S_n^+, U_0} \langle W, U \rangle$, где $U_{i,i} = 1$ для всех i (S_n^+ - обозначение для положительно полуопределенной матрицы), так как

- (1) матрица Грама положительно полуопределена
- (2) если $U \in S_n^+$, то $\exists X : U = X^T X$, то есть U -матрица скалярных произведений столбцов в X

Эта задача является задачей полуопределенного программирования и может быть решена за полиномиальное время. Притом заметим, что если все y_i имеют вид $(\pm 1, 0, \dots, 0)$, то множество

значений (3) совпадает с множеством значений (2), поэтому минимум функции (3) \leq (2), то есть максимум функции (1) не больше аналогичной для y_i . (4)

Получение разбиения из решения задачи оптимизации

Сначала получим векторы назад из матрицы скалярных произведений, то есть найдем $X : X^T X = U$. Это разложение может быть найдено за полиномиальное время.

Теперь мы хотим восстановить из матрицы X разрез. Рассмотрим столбцы этой матрицы y_1, \dots, y_n . Хотелось бы, чтобы алгоритм работал так, чтобы находящиеся "далеко" друг от друга векторы оказались в разных группах разреза. Для этого проведем через 0 случайную плоскость (выбранную из равномерного распределения плоскостей), и вершины, соответствующие номерам векторов в одной полуплоскости, отнесем при разрезе в первую долю, а остальные - во вторую.

Чтобы провести случайную плоскость, случайно и равномерно выберем $r \in S^{n-1}$ и построим плоскость, заданную уравнением $\langle r, x \rangle = 0$. Тогда, если $\langle r, y_i \rangle \geq 0$, то i -ю вершина в первой доле разреза, иначе во второй.

2.3 Алгоритм Геманса и Вильямсона. Оценка работы

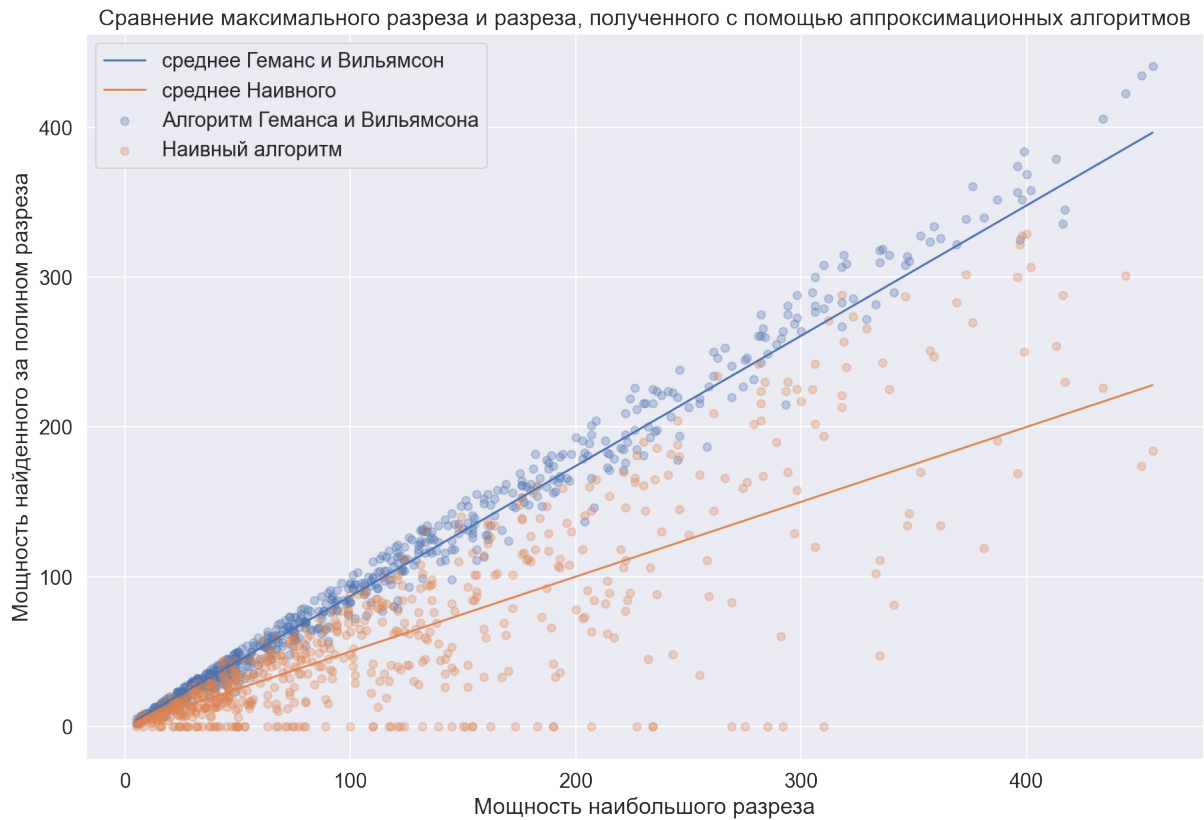
Лемма 1 Вероятность того, что случайная гиперплоскость $H : \langle r, x \rangle = 0$ разделит вектора y_i и y_j $\frac{\theta_{ij}}{2\pi}$ where θ_{ij} - угол между векторами y_i и y_j .

Proof. Пусть $J_{i,j}$ = поверхность, образованная векторами y_i и y_j (плоскость или прямая). $P(H \text{ разделит } i \text{ и } j) = P(\langle r, y_i \rangle \geq 0 \text{ и } \langle r, y_j \rangle < 0) + P(\langle r, y_j \rangle \geq 0 \text{ и } \langle r, y_i \rangle < 0) = P(H \cup J_{i,j} \text{ } y_i \text{ } y_j) = \frac{\theta_{ij}}{2\pi}$ \square

Лемма 2(б/д) $\frac{4}{\pi} \frac{\theta_{ij}}{(2\sin(\theta_{ij}/2))^2} > \alpha \approx 0.87856r0 \leq \theta_{ij} \leq \pi$

$E(\text{мощность разреза}) = \sum_{i=0, j=0}^{n,n} w_{i,j} * P(i \text{ и } j \text{ в разных долях}) = \sum_{i=0, j=0}^{n,n} w_{i,j} * \frac{\theta_{ij}}{4\pi} = \sum_{i=0, j=0}^{n,n} w_{i,j} * \frac{4}{\pi} \frac{\theta_{ij}}{(2\sin(\theta_{ij}/2))^2} \frac{\langle y_i - y_j, y_i - y_j \rangle}{8} \geq \alpha \sum_{i=0, j=0}^{n,n} \frac{\langle y_i - y_j, y_i - y_j \rangle}{8} = \alpha \sum_{i=0, j=0}^{n,n} \frac{\langle y_i - y_j, y_i - y_j \rangle}{8}$, что не меньше α * мощность максимального разреза (по соображению (4)). Таким образом, алгоритм Геманса и Вильямсона даёт аппроксимационный коэффициент около 87%

2.4 Сравнение наивного алгоритма и алгоритма Геманса и Вильямсона



Данный график визуализирует отношение ответа на задачу и реально полученного. На нем наглядно наблюдается, что аппроксимационный коэффициент 0,87 сильно лучше коэффициента 0,5. Так же на нем видно, что дисперсия алгоритма Геманса и Вильямсона достаточно маленькая, особенно по сравнению с наивным алгоритмом. Это говорит о том, что алгоритма Геманса и Вильямсона довольно точно приближает максимальный разрез.