Московский физико-технический институт

Поиск максимального разреза

Проект по сложности вычислений

Автор:
Седова Анна

Contents

Chapter 1

$\mathcal{NP}$-полнота

Задача MAXCUT

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

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 сравнивать с k, то верификатор будет работать за полиномиальное время.

□

MAXCUT является $\mathcal{NP}$-трудной

Proof. **1) PARTITION**

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

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

2) Покажем, что PARTITION $\leq_p$ 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 $\leq \sum_{i \in S_1, j \in S_2} c_i c_j = = \sum_{i \in S_1} c_i * \sum_{j \in S_2} c_j \leq (\frac{1}{2}\sum_{i=0}^{n} c_i)^2 = \frac{1}{4}\sum_{i=1}^{n} c_i^2$, то есть ровно $\frac{1}{4}\sum_{i=1}^{n} c_i^2$. Тогда в неравенстве $\sum_{i \in S_1} c_i * \sum_{j \in S_2} c_j \leq (\frac{1}{2}\sum_{i=0}^{n} c_i)^2$ достигается равенство, то есть $\sum_{i \in S_1} c_i = \sum_{j \in S_2} c_j = \frac{1}{2}\sum_{i=0}^{n} c_i$, 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 $\leq (\frac{1}{2}\sum_{i=0}^{n} c_i)^2 = \frac{1}{4}\sum_{i=1}^{n} c_i^2$, 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(мощность разреза) $= \sum_{i=0,j=0}^{n,n} w_{i,j} * P(i$ и j в разных долях$) = \sum_{i=0,j=0}^{n,n} w_{i,j} * 0.5 \geq 0.5*$ мощность максимального разреза. Таким образом, наивный алгоритм дает аппроксимационный коэффицент 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 множесва 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 $i = 1...n$. Тогда мощность этого разреза будет равна $|W| = \frac{1}{8}\sum_{i=0,j=0}^{n,n} w_{i,j} * (x_i - x_j)^2 = \frac{1}{8}\sum_{i=0,j=0}^{n,n} w_{i,j} * (x_i^2 + x_j^2 - 2x_ix_j)$. Заметим, что $x_i^2 = 1$, то есть $|W| = \frac{1}{8}\sum_{i=0,j=0}^{n,n} w_{i,j} * (2 - 2x_ix_j) = \frac{1}{4}\sum_{i=0,j=0}^{n,n} w_{i,j} * (1 - x_ix_j)$ **(1)**

Таким образом, поиск максимального разреза сводится к следующей задаче оптимизации $max_{x_i \in Z}\ \frac{1}{4}\sum_{i=0,j=0}^{n,n} w_{i,j} * (1 - x_ix_j)$, где $x_i^2 = 1$, что эквивалентно $min_{x_i \in Z} \sum_{i=0,j=0}^{n,n} w_{i,j} * (x_ix_j)$ **(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: $min_{y_i \in R^n} \sum_{i=0,j=0}^{n,n} w_{i,j} * < y_i, y_j >$ **(3)**, где $< y_i, y_i >= 1$, что эквивалентно $min_{U \in S_n^+, U0}$ <W, U> , 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 $(\pm 1, 0, ...0)$, то множество значений **(3)** совпадает с множеством значений **(2)**, поэтому минимум функции **(3)** $<=$ **(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 $y_1, ..., y_n$. 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 $< r, x >= 0$. Тогда, если $< r, y_i >\geq 0$, то i-ю вершина в первой доле разреза, иначе во второй.

2.3 Алгоритм Геманса и Вильямсона. Оценка работы

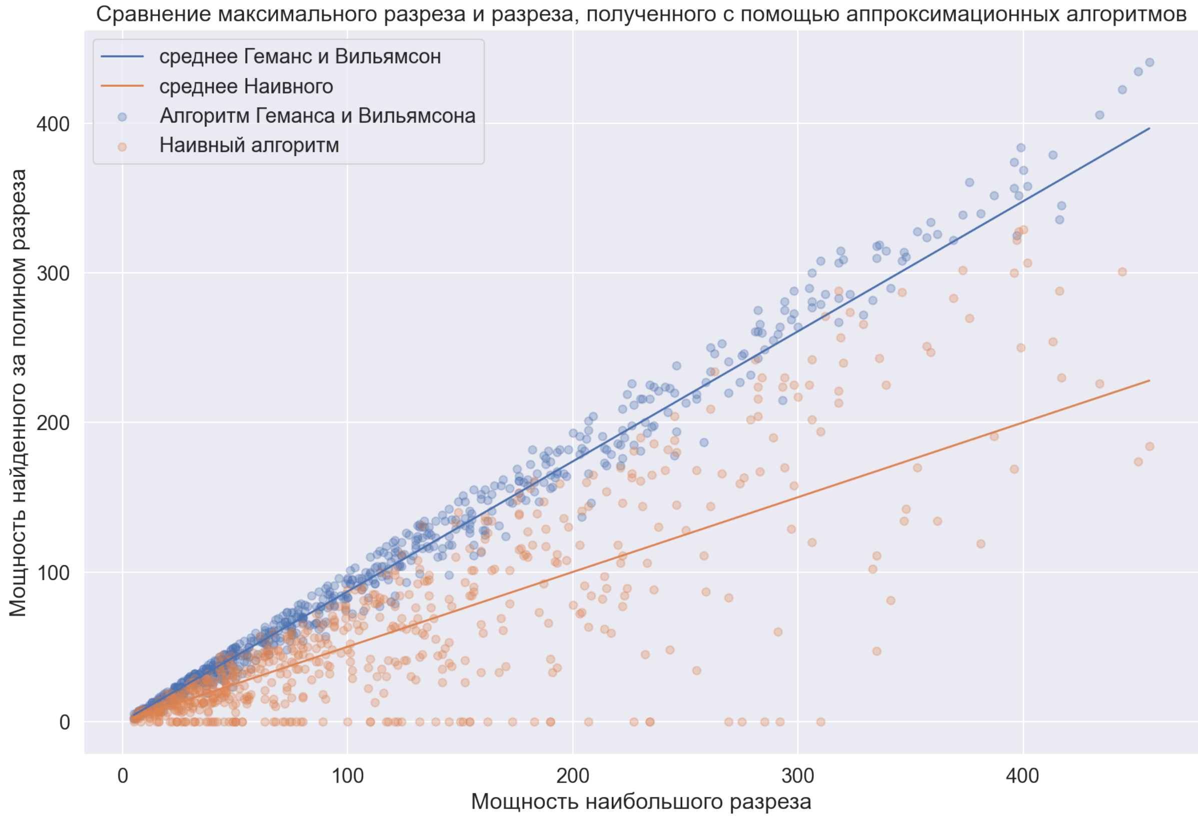
Лемма 1 Вероятность того, что случайная гиперплоскость $H :< r, x >= 0$ разделит вектора y_i и y_j $\frac{\theta_{ij}}{2\pi}$ where θ_{ij} - угол между векторами y_i и y_j.

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

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

E(мощность разреза) = $\sum_{i=0,j=0}^{n,n} w_{i,j} * P(i$ и j в разных долях$) = \sum_{i=0,j=0}^{n,n} w_{i,j} * \frac{\theta_{ij}}{4\pi} = \sum_{i=0,j=0}^{n,n} w_{i,j} * \quad * \frac{4}{\pi}\frac{\theta_{ij}}{(2sin(\theta_{ij}/2)^2}\frac{<y_i - y_j, y_i - y_j>}{8} \geq \alpha \sum_{i=0,j=0}^{n,n} w_{i,j}\frac{<y_i - y_j, y_i - y_j>}{8}$, что не меньше $\alpha *$ 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 коэффицент 0,87 сильно лучше коэффицента 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.

Chapter 3

Литература

1. Wikipedia

2. Michel X. Goemans and David P. Williamson, Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, Journal of the ACM (JACM), 42, 1115–1145, 1995.

3. UNIVERSITY OF WISCONSIN–MADISON Lectures Notes

4. Richard M. Karp (1972). "Reducibility Among Combinatorial Problems". In R. E. Miller and J. W. Thatcher (editors). Complexity of Computer Computations. New York: Plenum. pp. 85–103.