Калмыков Василий, 794
Алгоритм Гёманса-Уильямсона.

1 Постановка задачи

Дана 2-КНФ $\varphi(x), x = (x_1, \ldots, x_n)$, требуется найти такой набор значений переменных v, при котором выполняются как можно больше конъюнктов.

2 Сведение к задаче оптимизации

Заведём переменную $y_0 \in \{-1, 1\}$. Для каждой переменной v_i заведём переменную

$$y_i = \begin{cases} y_0, & \text{если } x_i = True \\ -y_0, & \text{если } x_i = False \end{cases}$$

Пусть C - логическая формула и $v(C) = 1$, если $C = True$, и $v(C) = 0$ при $C = False$. Тогда получаем, что

$$v(x_i) = \frac{1 + y_0 y_1}{2}$$

и

$$v(\overline{x}_i) = 1 - v(x_i) = \frac{1 - y_0 y_1}{2}$$

Теперь осталось выразить три варианта дизъюнктов:

$$v(x_i \vee x_j) = 1 - v(x_i \wedge x_j) = 1 - v(x_i) \cdot v(x_j) = 1 - \frac{1 - y_0 y_i}{2} \cdot \frac{1 - y_0 y_j}{2}$$

$$= \frac{1}{4}\left(3 + y_0 y_i + y_0 y_j - y_0^2 y_i y_j\right) = \frac{1 + y_0 y_i}{4} + \frac{1 + y_0 y_j}{4} + \frac{1 - y_i y_j}{4}$$

Заметим, что $v(\overline{x}_i)$ получается из $v(x_i)$ при помощи подстановки $-y_i$ вместо y_i. Таким образом, можем сразу получить остальные варианты дизъюнктов:

$$v(x_i \vee \overline{x}_j) = \frac{1 + y_0 y_i}{4} + \frac{1 - y_0 y_j}{4} + \frac{1 + y_i y_j}{4}$$

$$v(\overline{x}_i \vee x_j) = \frac{1 - y_0 y_i}{4} + \frac{1 + y_0 y_j}{4} + \frac{1 + y_i y_j}{4}$$

$$v(\overline{x}_i \vee \overline{x}_j) = \frac{1 - y_0 y_i}{4} + \frac{1 - y_0 y_j}{4} + \frac{1 - y_i y_j}{4}$$

Получаем, что нам нужно максимизировать следующее:

$$f(y) = \sum_{i<j} \left[a_{ij}(1 - y_i y_j) + b_{ij}(1 + y_i y_j)\right]$$

$a_{ij} =$ количество дизъюнктов с переменными x_i, x_j, в которых ровно 1 отрицание;

$a_{0i} =$ количество дизъюнктов с $\overline{x}_i$;

$b_{ij} =$ количество дизъюнктов с переменными x_i, x_j, в которых 0 или 2 отрицания;

$b_{0i} =$ количество дизъюнктов с x_i

3 Решение задачи поиска максимума

Преобразуем целевую функцию к более удобному виду:

$$f(y) = \sum_{i<j} [y_i y_j (b_{ij} - a_{ij})] + \sum_{i<j} [a_{ij} + b_{ij}]$$

Поскольку вторая сумма - константа, то максимизировать нужно лишь первую сумму. Получаем следующую задачу:

$$\max f(y) = \sum_i \sum_j C_{ij} y_i y_j$$

где

1. $C_{ij} = I\{i < j\} \cdot (b_{ij} - a_{ij})$;
2. $y = (y_0, \dots, y_n)$;
3. $y_i \in \{-1, 1\}$.

Если ослабить эту задачу, а именно позволить y_i быть вектором на единичной сфере в n-мерном пространстве:

$$\max f(y) = \sum_i \sum_j C_{ij} \langle y_i, y_j \rangle$$

где

1. $C_{ij} = I\{i < j\} \cdot (b_{ij} - a_{ij})$;
2. $y = (y_0, \dots, y_n)$;
3. $y \in S_{n+1} \subset \mathrm{R}^{n+1} \iff \|y\|_2 = 1$.

Это задача полуопределённого программирования.
(https://ru.wikipedia.org/wiki/Полуопределённое_программирование)
Решением этой задачи является набор векторов единичной длины. Осатнется лишь округлить их как-то, т.к. изначально были числа ± 1. Предлагается следующий вариант: равномерно случайно выбираем гиперплоскость в n-мерном пространстве, проходящую через 0. Эта плоскость разделит пространство на 2 части. Те вектора, что оказались по одну сторону от гиперплоскости, будем считать единицами. Остальные - минус единицами.
Альтернативное понимание: равномерно случайно выбираем $r \in S_{n+1}$, далее пусть $\{y_i\}$ - решения задачи SDP, а $\{b_i\}$ - решение исходной задачи(набор булевых переменных). Тогда $b_i := (\langle r, y_i \rangle >= 0)$. Правда, $\{b_i\}$ может оказаться не максимальным выполняющим набором, а противоположным ему (Это объясняется тем, что изначально это решение было придумано для задачи поиска максимального разреза, а в ней нужно просто разбить вершины на 2 группы, а в нашей задаче надо ещё и указать, какая группа - $True$). Потому в качестве ответа надо брать или $\{b_i\}$, или $\{\neg b_i\}$.

4 Анализ алгоритма

Теорема 1. *Обозначим за* $\mathbf{E}[\varphi(b)]$ *матожидание значения* φ *при наборе* b*, полученном указанным ранее способом.*

$$\mathbf{E}[\varphi(b)] = \frac{1}{\pi}\sum_{i<j} C_{ij} arccos(y_i \cdot y_j),\ \textit{где}\ \cdot\ \textit{- скалярное произведение}$$

Поскольку вектор r - равномерно распределён на единичной сфере S_{n+1}, то по линейности матожидания мы получаем:

$$\mathbf{E}\left[\varphi(b)\right] = \sum_{i<j} C_{ij} \cdot \mathbf{P}\left[\mathbf{sign}(y_i \cdot r) \neq \mathbf{sign}(y_j \cdot r)\right]$$

Следовательно, для доказательства теоремы надо доказать следующий факт:

Лемма 1.

$$\mathbf{P}\left[\mathbf{sign}(y_i \cdot r) \neq \mathbf{sign}(y_j \cdot r)\right] = \frac{arccos(y_i \cdot y_j)}{\pi}$$

Доказательство. Файтически нужно доказать, что вероятность того, что случайная гиперплоскость разделит конкретные 2 вектора, пропорциональна углу $\theta = \arccos(y_i \cdot y_j)$ между векторами. $\mathbf{P}\left[\mathbf{sign}(y_i \cdot r) \neq \mathbf{sign}(y_j \cdot r)\right] = \mathbf{P}\left[y_i \cdot r \geq 0, y_j \cdot r < 0\right] + \mathbf{P}\left[y_i \cdot r < 0, y_j \cdot r \geq 0\right]$. В силу симметрии: $\mathbf{P}\left[\mathbf{sign}(y_i \cdot r) \neq \mathbf{sign}(y_j \cdot r)\right] = 2\mathbf{P}\left[y_i \cdot r \geq 0, y_j \cdot r < 0\right]$. Множество $\{r : y_i \cdot r \geq 0, y_j \cdot r < 0\}$ - пересечение двух полупространств, и двугранный угол между этими полупространствами есть θ. Тогда вероятность попадания вектора r в это множество(т.е. $\mathbf{P}\left[y_i \cdot r \geq 0, y_j \cdot r < 0\right]$) равна $\theta/2pi$. Получаем, что

$$\mathbf{P}\left[\mathbf{sign}(y_i \cdot r) \neq \mathbf{sign}(y_j \cdot r)\right] = \frac{\arccos(y_i \cdot y_j)}{\pi}$$

□

Определим

$$\alpha = \min_{0 \leq \theta \leq \pi} \frac{2}{\pi} \frac{\theta}{1 - \cos\theta}$$

Теорема 2.

$$\mathbf{E}[\varphi(b)] \geq \frac{\alpha}{2} \sum_{i<j} C_{ij}(1 - y_i \cdot y_j)$$

Это основная теорема. Можно показать, что $\alpha \geq 0.87856$, и тогда получим, что матожидание результата составляет хотя бы 0.878 от *максимально возможного* результата.

Лемма 2.

$$\forall x \in [-1, 1] : \frac{arccos(x)}{\pi} \geq \alpha \cdot \frac{1}{2}(1 - x)$$

Лемма 3. $\alpha \geq 0.87856$

5 Детали реализации

Единичный вектор нормы 1 может быть сгенерирован так:

1. Генерируем стандартные нормальные с.в.;

2. Нормируем вектор.

(Авторы статьи ссылаются в этом месте на другую статью). Будем использовать библиотеку **cvxopt** для **python3**. Поскольку в этой библиотеке задача SDP выглядит как $minCX$, где X - симметричная положительно определённая матрица, то представим нашу задачу в таком виде.

$$\sum_i \sum_j C_{ij} \langle y_i, y_j \rangle = CY^TY, \text{ где } Y \text{ - матрица, столбцы которой - вектора.}$$

Т.е. в качестве ответа мы получим $Y^TY = X$. Далее можно воспользоваться декомпозицией Холецкого для того, чтобы получить уже Y - матрицу, столбцы которой - искомые вектора.