Алгоритм Кармаркара

Роман Сатюков.*

13 октября 2005 г.

План лекции

1. Формулировка задачи линейного программирования.
2. Вспомогательные утверждения.
3. Идея алгоритма.
4. Общая схема алгоритма.

1 Формулировка задачи линейного программирования

Пусть даны линейная функция $f(x)$ и система линейных неравенств $Ax \le b$. При этом пусть $\forall i\ x_i \ge 0$.

$$f(x) = c_1x_1 + c_2x_2 + \ldots + c_nx_n = \sum_{i=1}^{n} c_ix_i$$

$$\begin{cases} a_{11}x_1 + a_{12}x_1 + & \ldots + a_{1n}x_1 \le & b_1 \\ a_{21}x_1 + a_{22}x_2 + & \ldots + a_{2n}x_2 \le & b_2 \\ \vdots & \ddots & \vdots \\ a_{m1}x_1 + a_{m2}x_2 + & \ldots + a_{mn}x_n \le & b_n \end{cases}$$

Задача нахождения максимального значения функции при заданных линейных ограничениях называется задачей линейного программирования.

*конспектировал В.Данилов

2 Вспомогательные утверждения

2.1 Проекция вектора на линейное подпространство

Пусть $K = \{x \in R^n \mid Bx = 0\}$ - линейное подпространство, а c - произвольный вектор. Найдем c_p - проекцию C на K.

$$\begin{cases} b_{11}x_1 + b_{12}x_2 + & \ldots + b_{1n}x_n = & 0 \\ \vdots & \ddots & \vdots \\ b_{m1}x_1 + b_{m2}x_2 + & \ldots + b_{mn}x_n = & 0 \end{cases}$$

$$(c - c_p \perp K)$$

Таким образом

$$c - c_p = y_1 \begin{pmatrix} b_{11} \\ \vdots \\ b_{1n} \end{pmatrix} + \ldots + y_m \begin{pmatrix} b_{m1} \\ \vdots \\ b_{mn} \end{pmatrix}$$

$$c - c_p = B^T y$$
$$Bc - Bc_p = BB^T y$$
$$Bc = (BB^T)y$$

Здесь мы воспользовались тем, что $c_p \in K$, следовательно $Bc_p = 0$. Выражая из последнего уравнения y, получим оператор проектирования в явном виде:

$$\boxed{c_p = [E - B^T(BB^T)^{-1}]c}$$

2.2 Радиус вписанной и описанной сферы симплекса

Симплексом называется $\{(x1, \ldots, x2) \in R^n \mid \forall i \; x_i \geq 0, \sum_{i=1}^n x_i = 1\}$. Введем понятие грани симплекса - будем называть гранью симплекса множество точек симплекса, таких что $x_i = 0$. Вписанная сфера симплекса - это сфера, касающаяся всех граней симплекса.

Рассмотрим точку $(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n})$. Найдем расстояние до грани $x_i = 0$, используя теорему Пифагора - через расстояние от точки до плоскости $x_i = 0$ и расстояние от основания перпендикуляра до грани симплекса. Воспользуемся тем, что расстояние от точки x_0 до линейного подпространства $ax = b$ в R^n равно $\|ax_0 - b\|/\|a\|$. В нашем случае $a = (1, 1, \ldots, 1)$ и $b = 1$, поэтому расстояние от точки $(\frac{1}{n}, \frac{1}{n}, \ldots, 0, \frac{1}{n}, \ldots, \frac{1}{n})$ до подпространства будет равно $\frac{\frac{1}{n}}{\sqrt{\frac{n-1}{n}}}$.Таким образом расстояние до грани будет

$$\boxed{r = \frac{1}{\sqrt{n(n-1)}}}$$

Из произвольности i, следует что расстояние до любой грани симплекса будет равно r, следовательно рассмотренная точка является центром вписанной сферы симплекса, а радиус вписанной сферы равен r.

Радиус описанной сферы равен $\sqrt{\frac{n}{n-1}}$, а ее центр совпадает с центром вписанной сферы.

3 Идея алгоритма

Рассмотрим следующий частный случай задачи линейного программирования - необходимо минимизировать целевую функцию $f(x) = c^T x$, при следующих условиях

$$\begin{cases} Ax = 0 \\ \sum_{i=1}^{n} x_i = 1 \\ \forall i \ x_i \geq 0 \\ minc^T x = 0 \end{cases}$$

Потребуем также, что мы знаем некоторое точку a_0, удовдетворяющую системе.

Эту задачу можно решать следующим способом - найдем некое преобразование ϕ, такое, что $c^T \phi(x) < c^T x$, и построим последовательность точек $\{x_i\}$:

$$x_i = \begin{cases} a_0, & i = 0 \\ \phi(x_{i-1}), & i > 0 \end{cases}$$

Данная последовательность будет сходится к решению задачи - некой вершине симплекса. При этом, в определенный момент значение целевой функции станет настолько мало, что можно будет однозначно определить вершину на симплексе, к которой наша последовательность точек сходится. Более формально

$$L = \log(1 + D_{max}) + \log(1 + \alpha)$$

где D_{max} - максимальный по модулю определитель квадратной подматрицы A, а α - максимальный по модулю элемент из B или C. Утверждается, что количество шагов алгоритма составляет $O(L)$, после чего можно будет однозначно определить точку. Это обоснованно тем, что можно оценить минимальное расстояние между двумя точками симплекса, и если мы приблизимся к какой-то точке симплекса на достаточно близкое растояние, то это будет означать, что последовательность сходится к этой точке.

Осталось найти преобразование ϕ. Для этого рассмотрим преобразование ψ_a :

$$\psi_a(x) = \left(\frac{\frac{x_1}{a_1}}{\sum_{i=1}^{n} \frac{x_i}{a_i}}, \ldots, \frac{\frac{x_n}{a_n}}{\sum_{i=1}^{n} \frac{x_i}{a_i}} \right)$$

Нетрудно видеть, что ψ_x переводит x в центр вписанной сферы симплекса. Кроме того, описанное пребразование переводит точки на симплексе в точки на симплексе.

Сопоставим вектору a преобразования ψ_a диагональную матрицу D.

$$\begin{pmatrix} a_1 & 0 & \dots & 0 \\ 0 & a_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_n \end{pmatrix}$$

Тогда преобразование можно записать в форме

$$x\prime = \frac{D^{-1}x}{e^T D^{-1} x}$$

а обратное преобразование

$$x = \frac{Dx\prime}{e^T Dx\prime}$$

Целевая функция же преобразуется по правилу

$$c\prime = Dc$$

Построим теперь преобразование ϕ. Применим сначала ψ_x, переводя точку x в центр симплекса. Теперь некоторым преобразованием уменьшим значение целевой функцию, затем полученную точку подвергнем преобразованию, обратному ψ_x. Определим теперь, как уменьшить целевую функцию.

Пусть матрица B - это матрица AD, дополненная рядом из единиц. Теперь опустим проекцию $c\prime$ на линейное подпространство $Bx = 0$, и сместимся по полученному вектору на расстояние меньшее r. В итоге мы перейдем в точку на симплексе, причем не нарушим системы линейных ограничений и уменьшим значение целевой функции.

Заметим, что наше преобразование меняет целевую функцию нелинейно. Существует потенциальная функция, которая при наличии решения уменьшается линейно. Поэтому мы либо за $O(L)$ итераций достигнем решения, либо на одной из итераций потенциальная функция недостаточно уменьшится - что будет означать, что решение недостижимо. Таким образом построен алгоритм проверки того, достижимо ли решение с заданным значением целевой функции.

4 Общая схема алгоритма

Покажем теперь как свести решение задачи линейного программирования к рассмотренному случаю.

$$Ax \leq b$$

$$\begin{cases} Ax + y = b \\ y \geq 0 \end{cases}$$

Добавим координаты в переменные, и сопоставив им нулевые коэффициенты в целевой функции получим задачу линейного программирования к задаче линейного программирования, где $Ax = b$.

Покажем, как добиться того, что $b = 0$. Каждое уравнение исходной системы мы можем записать в форме $\sum_{i=1}^{n}(a_{ji} - b_j)x_i = 0$ и преобразовать данным образом A.

Покажем, как найти начальную точку. Решим для этого задачу линейного програмирования, где $Ax+y \geq b$ и надо минимизировать y. Начальная точка для этой задачи - любой x, и какой-то большой y. Мы можем решить эту задачу нашим алгоритмом. Минимизировав y до 0, мы получим точку, удовлетворяющую системе исходной задачи.

Теперь можно описать схему общего алгоритма. Если мы знаем что целевая функция решения лежит в каких-то границах между l и r, мы можем проверить существует ли решение с целевой функцией между $l + \frac{r-l}{3}$ и $l + \frac{2(r-l)}{3}$. Для этого попытаемся найти нашим алгоритмом для частного случая решение со значением $l+\frac{r-l}{3}$, и со значением $l+\frac{2(r-l)}{3}$, при этом подкорректировав целевую функцию(чтобы искать проверить достижимость решения с оценкой 0). В результате проверок мы сократим интервал за $O(L)$ итераций хотя бы в $\frac{2}{3}$ раза. Когда интервал сузится настолько, что можно будет однозначно определить решение, алгоритм завершает работу.

Построенный алгоритм работает за $O(n^{3.5} \times L^2)$.