с/к “Эффективные алгоритмы”

Лекция 17: Полиномиальный алгоритм для задачи линейного программирования

А. Куликов

Computer Science клуб при ПОМИ

http://logic.pdmi.ras.ru/∼infclub/

План лекции

1. Введение

План лекции

План лекции

1. Введение
2. Масштабирование и направление скорейшего спуска
3. Постановка задачи, потенциал
   - Постановка задачи
   - Потенциал

План лекции

1. Введение
2. Масштабирование и направление скорейшего спуска
3. Постановка задачи, потенциал
   - Постановка задачи
   - Потенциал
4. Алгоритм Кармаркара

План лекции

План лекции

Общий вид задачи линейного программирования

Общий вид задачи линейного программирования

$$\begin{array}{rlcl} \text{минимизировать} & c^T x & & \\ \text{при условиях} & Ax & = & b \\ & x & \geq & 0, \end{array}$$

где $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, A — матрица полного ранга размера $m \times n$, $m < n$.

Общая идея симплекс-метода

Общая идея симплекс-метода

Общая идея симплекс-метода

Общая идея симплекс-метода

- Множество допустимых решений является выпуклым многогранником.

Общая идея симплекс-метода

Общая идея симплекс-метода

- Множество допустимых решений является выпуклым многогранником.
- Симплекс-метод переходит от одной вершины этого многогранника к другой до тех пор, пока это улучшает значение целевой функции.

Общая идея симплекс-метода

Общая идея симплекс-метода

- Множество допустимых решений является выпуклым многогранником.
- Симплекс-метод переходит от одной вершины этого многогранника к другой до тех пор, пока это улучшает значение целевой функции.
- Известны примеры, на которых симплекс-метод будет работать экспоненицально долго (такие примеры, однако, не встречаются на практике).

Общая идея метода внутренней точки

Общая идея метода внутренней точки

Общая идея метода внутренней точки

Общая идея метода внутренней точки

- Выбирается произвольная точка x^0 внутри многогранника допустимых решений.

Общая идея метода внутренней точки

Общая идея метода внутренней точки

- Выбирается произвольная точка x^0 внутри многогранника допустимых решений.
- Найдем преобразование ϕ, для которого $c^T\phi(x) < c^T x$.

Общая идея метода внутренней точки

Общая идея метода внутренней точки

- Выбирается произвольная точка x^0 внутри многогранника допустимых решений.
- Найдем преобразование ϕ, для которого $c^T\phi(x) < c^Tx$.
- На каждой итерации переходим от x^k к $x^{k+1} = \phi(x^k)$, оставаясь внутри многогранника.

Общая идея метода внутренней точки

Общая идея метода внутренней точки

- Выбирается произвольная точка x^0 внутри многогранника допустимых решений.
- Найдем преобразование ϕ, для которого $c^T\phi(x) < c^T x$.
- На каждой итерации переходим от x^k к $x^{k+1} = \phi(x^k)$, оставаясь внутри многогранника.
- Когда значение $c^T\phi(x^k)$ станет достаточно маленьким, перейдём из текущей точки в ближайшую вершину многогранника, которая и будет решением.

Основные составляющие алгоритма Кармаркара

Основные составляющие

Основные составляющие алгоритма Кармаркара

Основные составляющие

1. масштабирование (scaling)

Основные составляющие алгоритма Кармаркара

Основные составляющие

1. масштабирование (scaling)
2. направление быстрейшего спуска (steepest descent direction)

Основные составляющие алгоритма Кармаркара

Основные составляющие

1. масштабирование (scaling)
2. направление быстрейшего спуска (steepest descent direction)
3. потенциал (potential function)

План лекции

1. Введение
2. Масштабирование и направление скорейшего спуска
3. Постановка задачи, потенциал
   - Постановка задачи
   - Потенциал
4. Алгоритм Кармаркара
5. Анализ алгоритма

Обобщенная задача

Обобщенная задача

$$\begin{array}{rlcl} \text{минимизировать} & f(x) & & \\ \text{при условиях} & Ax & = & b \\ & x & \geq & 0, \end{array}$$

где $f : \mathbb{R}^n_{\geq 0} \to \mathbb{R}$, $b \in \mathbb{R}^m$, A — матрица полного ранга размера $m \times n$, $m < n$.

Общие идеи алгоритма

Общие идеи алгоритма

Общие идеи алгоритма

Общие идеи алгоритма

- Мы представим для этой задачи алгоритм, основанный на методе скорейшего спуска, в начале каждой итерации которого производится масштабирование.

Общие идеи алгоритма

Общие идеи алгоритма

- Мы представим для этой задачи алгоритм, основанный на методе скорейшего спуска, в начале каждой итерации которого производится масштабирование.
- Каждая итерация начинается с точки x^k, а масштабирование выглядит следующим образом: $x \to y = X^{-1}x$, где $X = \mathrm{diag}(x_1^k, \ldots, x_n^k)$ (таким образом, x^k переходит в e).

Алгоритм

Алгоритм

Допустим, что дана исходная допустимая точка $x^0 > 0$ и число $\delta \in (0, 1]$. Положим $k = 0$ и будем повторять (до некоторого момента) следующие шаги:

Алгоритм

Алгоритм

Допустим, что дана исходная допустимая точка $x^0 > 0$ и число $\delta \in (0,1]$. Положим $k = 0$ и будем повторять (до некоторого момента) следующие шаги:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \ldots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{f}(y) \leftarrow f(Xy)$.

Алгоритм

Алгоритм

Допустим, что дана исходная допустимая точка $x^0 > 0$ и число $\delta \in (0, 1]$. Положим $k = 0$ и будем повторять (до некоторого момента) следующие шаги:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \ldots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{f}(y) \leftarrow f(Xy)$.
- Направление: $h \leftarrow -P\nabla\bar{f}(e)$, где $P \leftarrow I - \bar{A}^T \left(\bar{A}\bar{A}^T\right)^{-1} \bar{A}$ — проектор на ядро $\mathrm{Null}(\bar{A})$.

Алгоритм

Алгоритм

Допустим, что дана исходная допустимая точка $x^0 > 0$ и число $\delta \in (0, 1]$. Положим $k = 0$ и будем повторять (до некоторого момента) следующие шаги:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \ldots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{f}(y) \leftarrow f(Xy)$.
- Направление: $h \leftarrow -P\nabla\bar{f}(e)$, где $P \leftarrow I - \bar{A}^T \left(\bar{A}\bar{A}^T\right)^{-1} \bar{A}$ — проектор на ядро $\mathrm{Null}(\bar{A})$.
- Поиск на прямой: найти $\bar{\lambda} \in \mathrm{argmin}\{\bar{f}(e + \lambda h) | \lambda \geq 0, e + \lambda h \geq 0\}$.

Алгоритм

Алгоритм

Допустим, что дана исходная допустимая точка $x^0 > 0$ и число $\delta \in (0, 1]$. Положим $k = 0$ и будем повторять (до некоторого момента) следующие шаги:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \ldots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{f}(y) \leftarrow f(Xy)$.
- Направление: $h \leftarrow -P\nabla\bar{f}(e)$, где $P \leftarrow I - \bar{A}^T\left(\bar{A}\bar{A}^T\right)^{-1}\bar{A}$ — проектор на ядро $\mathrm{Null}(\bar{A})$.
- Поиск на прямой: найти $\bar{\lambda} \in \mathrm{argmin}\{\bar{f}(e + \lambda h) | \lambda \geq 0, e + \lambda h \geq 0\}$.
- Результат: $y^* \leftarrow e + \delta\bar{\lambda}h$.

Алгоритм

Алгоритм

Допустим, что дана исходная допустимая точка $x^0 > 0$ и число $\delta \in (0, 1]$. Положим $k = 0$ и будем повторять (до некоторого момента) следующие шаги:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \ldots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{f}(y) \leftarrow f(Xy)$.
- Направление: $h \leftarrow -P\nabla\bar{f}(e)$, где $P \leftarrow I - \bar{A}^T \left(\bar{A}\bar{A}^T\right)^{-1} \bar{A}$ — проектор на ядро $\mathrm{Null}(\bar{A})$.
- Поиск на прямой: найти $\bar{\lambda} \in \mathrm{argmin}\{\bar{f}(e + \lambda h) | \lambda \geq 0, e + \lambda h \geq 0\}$.
- Результат: $y^* \leftarrow e + \delta\bar{\lambda}h$.
- Обратно: $x^{k+1} \leftarrow Xy^*$.

Алгоритм

Алгоритм

Допустим, что дана исходная допустимая точка $x^0 > 0$ и число $\delta \in (0,1]$. Положим $k = 0$ и будем повторять (до некоторого момента) следующие шаги:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \dots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{f}(y) \leftarrow f(Xy)$.
- Направление: $h \leftarrow -P\nabla\bar{f}(e)$, где $P \leftarrow I - \bar{A}^T \left(\bar{A}\bar{A}^T\right)^{-1} \bar{A}$ — проектор на ядро $\mathrm{Null}(\bar{A})$.
- Поиск на прямой: найти $\bar{\lambda} \in \mathrm{argmin}\{\bar{f}(e+\lambda h) | \lambda \geq 0, e + \lambda h \geq 0\}$.
- Результат: $y^* \leftarrow e + \delta\bar{\lambda}h$.
- Обратно: $x^{k+1} \leftarrow Xy^*$.
- $k \leftarrow k + 1$.

План лекции

План лекции

Вид задачи

Вид задачи

$$\begin{array}{rlcl} \text{минимизировать} & c^T x & & \\ \text{при условиях} & Ax & = & 0 \\ & x & \geq & 0, \end{array}$$

где $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, A — матрица полного ранга размера $m \times n$, $m < n$.

Дополнительные предположения

Дополнительные предположения

Дополнительные предположения

Дополнительные предположения

- Множество допустимых решений $S = \{x \in \mathbb{R}_+^n | Ax = 0\}$ ограничено.

Дополнительные предположения

Дополнительные предположения

- Множество допустимых решений $S = \{x \in \mathbb{R}^n_+ | Ax = 0\}$ ограничено.
- Точка $x^0 = e$ допустима. (Известны процедуры нахождения начальной допустимой точки x^0. Масштабированием такую точку можно перевести в e.)

Дополнительные предположения

Дополнительные предположения

- Множество допустимых решений $S = \{x \in \mathbb{R}^n_+ | Ax = 0\}$ ограничено.
- Точка $x^0 = e$ допустима. (Известны процедуры нахождения начальной допустимой точки x^0. Масштабированием такую точку можно перевести в e.)
- $c^T x$ не константа на S.

Дополнительные предположения

Дополнительные предположения

- Множество допустимых решений $S = \{x \in \mathbb{R}^n_+ | Ax = 0\}$ ограничено.
- Точка $x^0 = e$ допустима. (Известны процедуры нахождения начальной допустимой точки x^0. Масштабированием такую точку можно перевести в e.)
- $c^T x$ не константа на S.
- Значение любого оптимального решения $\hat{x}$ есть $c^T \hat{x} = 0$. (Очень сильное условие, от которого, тем не менее, можно избавиться.)

План лекции

Потенциал

Потенциал

$$x \in \mathbb{R}^n_{>0} \mapsto f(x) = n\log(c^Tx) - \sum_{i=1}^{n} \log x_i = \log \frac{(c^Tx)^n}{\prod_{i=1}^{n} x_i}.$$

Потенциал

Потенциал

$$x \in \mathbb{R}^n_{>0} \mapsto f(x) = n \log(c^T x) - \sum_{i=1}^{n} \log x_i = \log \frac{(c^T x)^n}{\prod_{i=1}^{n} x_i}.$$

Задача

Найти $x \in S$, такой что $f(x) < -2nL$.

Потенциал

Потенциал

$$x \in \mathbb{R}^n_{>0} \mapsto f(x) = n \log(c^T x) - \sum_{i=1}^{n} \log x_i = \log \frac{(c^T x)^n}{\prod_{i=1}^{n} x_i}.$$

Задача

Найти $x \in S$, такой что $f(x) < -2nL$.

Интуитивно

Мы хотим, чтобы первое слагаемое в определении x было бы как можно меньше. Второе же слагаемое (логарифмический барьер) неограниченно возрастает у границы $\mathbb{R}^n_{\geq 0}$ и ограничено снизу, поскольку S ограничено.

План лекции

Алгоритм Кармаркара

Алгоритм Кармаркара

Допустим, что даны $x^0 = e \in S$ и $L > 0$. Положим $k = 0$ и будем повторять:

Алгоритм Кармаркара

Алгоритм Кармаркара

Допустим, что даны $x^0 = e \in S$ и $L > 0$. Положим $k = 0$ и будем повторять:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \ldots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{c} \leftarrow Xc$.

Алгоритм Кармаркара

Алгоритм Кармаркара

Допустим, что даны $x^0 = e \in S$ и $L > 0$. Положим $k = 0$ и будем повторять:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \ldots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{c} \leftarrow Xc$.
- Направление: $h \leftarrow -\frac{n}{\bar{c}^T e} P\bar{c} + e$, где $P \leftarrow I - \bar{A}^T \left(\bar{A}\bar{A}^T\right)^{-1} \bar{A}$ — проектор на ядро $\mathrm{Null}(\bar{A})$.

Алгоритм Кармаркара

Алгоритм Кармаркара

Допустим, что даны $x^0 = e \in S$ и $L > 0$. Положим $k = 0$ и будем повторять:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \ldots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{c} \leftarrow Xc$.
- Направление: $h \leftarrow -\frac{n}{\bar{c}^T e} P\bar{c} + e$, где $P \leftarrow I - \bar{A}^T \left(\bar{A}\bar{A}^T\right)^{-1} \bar{A}$ — проектор на ядро $\mathrm{Null}(\bar{A})$.
- Поиск на прямой: найти $\bar{\lambda} \in \mathrm{argmin}\{\bar{f}(e + \lambda h) | \lambda \geq 0, e + \lambda h \geq 0\}$.

Алгоритм Кармаркара

Алгоритм Кармаркара

Допустим, что даны $x^0 = e \in S$ и $L > 0$. Положим $k = 0$ и будем повторять:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \ldots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{c} \leftarrow Xc$.
- Направление: $h \leftarrow -\frac{n}{\bar{c}^T e} P\bar{c} + e$, где $P \leftarrow I - \bar{A}^T \left(\bar{A}\bar{A}^T\right)^{-1} \bar{A}$ — проектор на ядро $\mathrm{Null}(\bar{A})$.
- Поиск на прямой: найти $\bar{\lambda} \in \mathrm{argmin}\{\bar{f}(e + \lambda h) | \lambda \geq 0, e + \lambda h \geq 0\}$.
- Результат: $y^* \leftarrow e + \bar{\lambda} h$.

Алгоритм Кармаркара

Алгоритм Кармаркара

Допустим, что даны $x^0 = e \in S$ и $L > 0$. Положим $k = 0$ и будем повторять:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \ldots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{c} \leftarrow Xc$.
- Направление: $h \leftarrow -\frac{n}{\bar{c}^T e} P\bar{c} + e$, где $P \leftarrow I - \bar{A}^T \left(\bar{A}\bar{A}^T\right)^{-1} \bar{A}$ — проектор на ядро $\mathrm{Null}(\bar{A})$.
- Поиск на прямой: найти $\bar{\lambda} \in \mathrm{argmin}\{\bar{f}(e + \lambda h) | \lambda \geq 0, e + \lambda h \geq 0\}$.
- Результат: $y^* \leftarrow e + \bar{\lambda} h$.
- Обратно: $x^{k+1} \leftarrow Xy^*$.

Алгоритм Кармаркара

Алгоритм Кармаркара

Допустим, что даны $x^0 = e \in S$ и $L > 0$. Положим $k = 0$ и будем повторять:

- Масштабирование: $X \leftarrow \mathrm{diag}(x_1^k, \dots, x_n^k)$, $\bar{A} \leftarrow AX$, $\bar{c} \leftarrow Xc$.
- Направление: $h \leftarrow -\frac{n}{\bar{c}^T e} P\bar{c} + e$, где $P \leftarrow I - \bar{A}^T \left(\bar{A}\bar{A}^T\right)^{-1} \bar{A}$ — проектор на ядро $\mathrm{Null}(\bar{A})$.
- Поиск на прямой: найти $\bar{\lambda} \in \mathrm{argmin}\{\bar{f}(e + \lambda h) | \lambda \geq 0, e + \lambda h \geq 0\}$.
- Результат: $y^* \leftarrow e + \bar{\lambda} h$.
- Обратно: $x^{k+1} \leftarrow X y^*$.
- $k \leftarrow k + 1$.

пока $f(x^k) \geq -2nL$.

План лекции

1. Введение
2. Масштабирование и направление скорейшего спуска
3. Постановка задачи, потенциал
   - Постановка задачи
   - Потенциал
4. Алгоритм Кармаркара
5. Анализ алгоритма

Важное свойство алгоритма (без доказательства)

Лемма

$$f(x^{k+1}) < f(x^k) - 0.1.$$

Важное свойство алгоритма (без доказательства)

Лемма

$$f(x^{k+1}) < f(x^k) - 0.1.$$

Замечания

Важное свойство алгоритма (без доказательства)

Лемма

$$f(x^{k+1}) < f(x^k) - 0.1.$$

Замечания

- Из этого следует, что не более чем через $30nL$ итераций потенциал уменьшится хотя бы на $3nL$.

Важное свойство алгоритма (без доказательства)

Лемма

$$f(x^{k+1}) < f(x^k) - 0.1.$$

Замечания

- Из этого следует, что не более чем через $30nL$ итераций потенциал уменьшится хотя бы на $3nL$.
- Мы также покажем, что такого уменьшения потенциала достаточно, чтобы найти оптимальное значение.

Исходная задача

Исходная задача

$$\begin{array}{rlcl} \text{минимизировать} & c^T x & & \\ \text{при условиях} & Ax & = & b \\ & x & \geq & 0, \end{array}$$

где $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, A — матрица полного ранга размера $m \times n$, $m < n$.

Исходная задача

Исходная задача

$$\begin{array}{rlcl} \text{минимизировать} & c^T x & & \\ \text{при условиях} & Ax & = & b \\ & x & \geq & 0, \end{array}$$

где $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, A — матрица полного ранга размера $m \times n$, $m < n$.

Замечание

Допустим, что многогранник допустимых решений ограничен и что все числа целые (или, что эквивалентно, рациональные).

Определение числа L

Определение числа L

Определение числа L

Определение числа L

- Пусть l равно общей длине битового представления входных данных.

Определение числа L

Определение числа L

- Пусть l равно общей длине битового представления входных данных.
- Заметим, что любое выражение, содержащее суммы и произведения чисел из входных матриц и векторов без повторений, не может иметь значение больше 2^l.

Определение числа L

Определение числа L

- Пусть l равно общей длине битового представления входных данных.
- Заметим, что любое выражение, содержащее суммы и произведения чисел из входных матриц и векторов без повторений, не может иметь значение больше 2^l.
- Определитель матрицы не превосходит произведения норм столбцов, поэтому $|\det M| \leq 2^l$ для любой квадратной подматрицы матрицы $[A|b]$.

Определение числа L

Определение числа L

- Пусть l равно общей длине битового представления входных данных.
- Заметим, что любое выражение, содержащее суммы и произведения чисел из входных матриц и векторов без повторений, не может иметь значение больше 2^l.
- Определитель матрицы не превосходит произведения норм столбцов, поэтому $|\det M| \leq 2^l$ для любой квадратной подматрицы матрицы $[A|b]$.
- Пусть $L = l + n + 1$.

Ключевая лемма

Лемма

Ключевая лемма

Лемма

- Для любой вершины x многогранника и для любого i либо $x_i = 0$, либо $x_i > 2^{-L}$.

Ключевая лемма

Лемма

- Для любой вершины x многогранника и для любого i либо $x_i = 0$, либо $x_i > 2^{-L}$.
- Для любой вершины x многогранника либо $c^T x = 0$, либо $c^T x > 2^{-L}$.

Ключевая лемма

Лемма

- Для любой вершины x многогранника и для любого i либо $x_i = 0$, либо $x_i > 2^{-L}$.
- Для любой вершины x многогранника либо $c^T x = 0$, либо $c^T x > 2^{-L}$.
- Для любой допустимой точки x $c^T x \leq 2^L$ и $e^T x \leq 2^L$.

Ключевая лемма

Лемма

- Для любой вершины x многогранника и для любого i либо $x_i = 0$, либо $x_i > 2^{-L}$.
- Для любой вершины x многогранника либо $c^T x = 0$, либо $c^T x > 2^{-L}$.
- Для любой допустимой точки x $c^T x \leq 2^L$ и $e^T x \leq 2^L$.

Доказательство

Ключевая лемма

Лемма

- Для любой вершины x многогранника и для любого i либо $x_i = 0$, либо $x_i > 2^{-L}$.
- Для любой вершины x многогранника либо $c^T x = 0$, либо $c^T x > 2^{-L}$.
- Для любой допустимой точки x $c^T x \leq 2^L$ и $e^T x \leq 2^L$.

Доказательство

- Система $Ax = b$ может быть разбита на $A_B x_B + A_N x_N = b$, где x_B — вектор, составленный из положительных компонент x.

Ключевая лемма

Лемма

- Для любой вершины x многогранника и для любого i либо $x_i = 0$, либо $x_i > 2^{-L}$.
- Для любой вершины x многогранника либо $c^T x = 0$, либо $c^T x > 2^{-L}$.
- Для любой допустимой точки x $c^T x \leq 2^L$ и $e^T x \leq 2^L$.

Доказательство

- Система $Ax = b$ может быть разбита на $A_B x_B + A_N x_N = b$, где x_B — вектор, составленный из положительных компонент x.
- Тогда $x_N = 0$ и система может быть переписана как $A_B x_B = b$.

Ключевая лемма

Лемма

- Для любой вершины x многогранника и для любого i либо $x_i = 0$, либо $x_i > 2^{-L}$.
- Для любой вершины x многогранника либо $c^T x = 0$, либо $c^T x > 2^{-L}$.
- Для любой допустимой точки x $c^T x \leq 2^L$ и $e^T x \leq 2^L$.

Доказательство

- Система $Ax = b$ может быть разбита на $A_B x_B + A_N x_N = b$, где x_B — вектор, составленный из положительных компонент x.
- Тогда $x_N = 0$ и система может быть переписана как $A_B x_B = b$.
- Если x — вершина, то столбцы A_B линейно независимы.

Доказательство (продолжение)

Доказательство

Доказательство (продолжение)

Доказательство

- Будем считать, что A_B — квадратная матрица (в противном случае, выкинем лишние уравнения).

Доказательство (продолжение)

Доказательство

- Будем считать, что A_B — квадратная матрица (в противном случае, выкинем лишние уравнения).
- Тогда $x_i = \Delta_i / \Delta$, где $\Delta = \det A_B$, а Δ_i — определитель матрицы, полученной заменой столбца A_i на b.

Доказательство (продолжение)

Доказательство

- Будем считать, что A_B — квадратная матрица (в противном случае, выкинем лишние уравнения).
- Тогда $x_i = \Delta_i/\Delta$, где $\Delta = \det A_B$, а Δ_i — определитель матрицы, полученной заменой столбца A_i на b.
- Пусть l_1, l_c суть количества битов в записи $\{A, b\}$ и c.

Доказательство (продолжение)

Доказательство

- Будем считать, что A_B — квадратная матрица (в противном случае, выкинем лишние уравнения).
- Тогда $x_i = \Delta_i/\Delta$, где $\Delta = \det A_B$, а Δ_i — определитель матрицы, полученной заменой столбца A_i на b.
- Пусть l_1, l_c суть количества битов в записи $\{A, b\}$ и c.
- Первое утверждение верно, поскольку Δ_i — целое, а $|\Delta| \le 2^{l_1} \le 2^L$.

Доказательство (продолжение)

Доказательство

- Будем считать, что A_B — квадратная матрица (в противном случае, выкинем лишние уравнения).
- Тогда $x_i = \Delta_i/\Delta$, где $\Delta = \det A_B$, а Δ_i — определитель матрицы, полученной заменой столбца A_i на b.
- Пусть l_1, l_c суть количества битов в записи $\{A, b\}$ и c.
- Первое утверждение верно, поскольку Δ_i — целое, а $|\Delta| \leq 2^{l_1} \leq 2^L$.
- Второе утверждение следует из первого и того факта, что c — целочисленный.

Доказательство (продолжение)

Доказательство

- Будем считать, что A_B — квадратная матрица (в противном случае, выкинем лишние уравнения).
- Тогда $x_i = \Delta_i/\Delta$, где $\Delta = \det A_B$, а Δ_i — определитель матрицы, полученной заменой столбца A_i на b.
- Пусть l_1, l_c суть количества битов в записи $\{A, b\}$ и c.
- Первое утверждение верно, поскольку Δ_i — целое, а $|\Delta| \leq 2^{l_1} \leq 2^L$.
- Второе утверждение следует из первого и того факта, что c — целочисленный.
- Для доказательства третьего утверждения рассмотрим вершину x многогранника.

Доказательство (завершение)

Доказательство

Доказательство (завершение)

Доказательство

- Тогда $e^T x \leq n2^{l_1} \leq 2^L$ и $c^T x \leq n2^{l_1+l_c} \leq 2^L$.

Доказательство (завершение)

Доказательство

- Тогда $e^T x \leq n2^{l_1} \leq 2^L$ и $c^T x \leq n2^{l_1+l_c} \leq 2^L$.
- Третье утверждение, таким образом, доказано для всех вершин многогранника.

Доказательство (завершение)

Доказательство

- Тогда $e^T x \leq n2^{l_1} \leq 2^L$ и $c^T x \leq n2^{l_1+l_c} \leq 2^L$.
- Третье утверждение, таким образом, доказано для всех вершин многогранника.
- Поскольку многогранник ограничен, любая его внутренняя точка может быть представлена в виде выпуклой комбинации его вершин.

Доказательство (завершение)

Доказательство

- Тогда $e^T x \leq n2^{l_1} \leq 2^L$ и $c^T x \leq n2^{l_1+l_c} \leq 2^L$.
- Третье утверждение, таким образом, доказано для всех вершин многогранника.
- Поскольку многогранник ограничен, любая его внутренняя точка может быть представлена в виде выпуклой комбинации его вершин.
- Итак, пусть $x = \sum_{j \in J} \lambda_j x^j$, где $x_j \geq 0$ и $\sum_{j \in J} \lambda_j = 1$.

Доказательство (завершение)

Доказательство

- Тогда $e^T x \leq n 2^{l_1} \leq 2^L$ и $c^T x \leq n 2^{l_1 + l_c} \leq 2^L$.
- Третье утверждение, таким образом, доказано для всех вершин многогранника.
- Поскольку многогранник ограничен, любая его внутренняя точка может быть представлена в виде выпуклой комбинации его вершин.
- Итак, пусть $x = \sum_{j \in J} \lambda_j x^j$, где $x_j \geq 0$ и $\sum_{j \in J} \lambda_j = 1$.
- Тогда $c^T x = \sum_{j \in J} \lambda_j c^T x^j \leq 2^L \sum_{j \in J} \lambda_j = 2^L$.

Доказательство (завершение)

Доказательство

- Тогда $e^T x \le n2^{l_1} \le 2^L$ и $c^T x \le n2^{l_1+l_c} \le 2^L$.
- Третье утверждение, таким образом, доказано для всех вершин многогранника.
- Поскольку многогранник ограничен, любая его внутренняя точка может быть представлена в виде выпуклой комбинации его вершин.
- Итак, пусть $x = \sum_{j \in J} \lambda_j x^j$, где $x_j \ge 0$ и $\sum_{j \in J} \lambda_j = 1$.
- Тогда $c^T x = \sum_{j \in J} \lambda_j c^T x^j \le 2^L \sum_{j \in J} \lambda_j = 2^L$.
- Оценка на $e^T x$ доказывается аналогично. □

Ещё один факт

Факт

Существует алгоритм, находящий по допустимой точке x за время $O(n^3)$ вершину многогранника $\bar{x}$, такую что

$$c^T\bar{x} \leq c^T x.$$

Основная теорема

Теорема

Алгоритм Кармаркара находит оптимальное значение за не более чем $O(nL)$ итераций (и $O(n^4L)$ арифметических операций).

Основная теорема

Теорема

Алгоритм Кармаркара находит оптимальное значение за не более чем $O(nL)$ итераций (и $O(n^4L)$ арифметических операций).

Доказательство

Основная теорема

Теорема

Алгоритм Кармаркара находит оптимальное значение за не более чем $O(nL)$ итераций (и $O(n^4L)$ арифметических операций).

Доказательство

- $f(x^0) = f(e) = n\log(c^T e) \leq n\log\left(2^L\right) \leq nL$

Основная теорема

Теорема

Алгоритм Кармаркара находит оптимальное значение за не более чем $O(nL)$ итераций (и $O(n^4 L)$ арифметических операций).

Доказательство

- $f(x^0) = f(e) = n \log(c^T e) \leq n \log\left(2^L\right) \leq nL$
- $f(x^k) < f(x^0) - 0.1k \leq nL - 0.1k$

Основная теорема

Теорема

Алгоритм Кармаркара находит оптимальное значение за не более чем $O(nL)$ итераций (и $O(n^4L)$ арифметических операций).

Доказательство

- $f(x^0) = f(e) = n\log(c^Te) \le n\log\left(2^L\right) \le nL$
- $f(x^k) < f(x^0) - 0.1k \le nL - 0.1k$
- Значит, через не более чем $30nL$ итераций алгоритм остановится на точке
$$f(x^k) = n\log(c^Tx^k) - \sum_{i=1}^{n}\log x_i^k < -2nL$$
.

Доказательство (завершение)

Доказательство

Доказательство (завершение)

Доказательство

- $\sum_{i=1}^{n} \log x_i^k < nL$, а значит, $\log(c^T x^k) < -L$.

Доказательство (завершение)

Доказательство

- $\sum_{i=1}^{n} \log x_i^k < nL$, а значит, $\log(c^T x^k) < -L$.
- Тогда $c^T x^k < e^{-L} < 2^{-L}$.

Доказательство (завершение)

Доказательство

- $\sum_{i=1}^{n} \log x_i^k < nL$, а значит, $\log(c^T x^k) < -L$.
- Тогда $c^T x^k < e^{-L} < 2^{-L}$.
- При нахождении такой точки идём от нее к вершине-решению. $\square$

Что мы узнали за сегодня?

Что мы узнали за сегодня?

Что мы узнали за сегодня?

Что мы узнали за сегодня?

- Симплекс-метод ходит по вершинам многоугольника допустимых решений, улучшая значение целевой функции. Время работы может быть экспоненицальным.

Что мы узнали за сегодня?

Что мы узнали за сегодня?

- Симплекс-метод ходит по вершинам многоугольника допустимых решений, улучшая значение целевой функции. Время работы может быть экспоненицальным.
- Метод внутренней точки тоже переходит от точки к точке, улучшая значение целевой функции, но остаётся при это во внутренности многогранника.

Что мы узнали за сегодня?

Что мы узнали за сегодня?

- Симплекс-метод ходит по вершинам многоугольника допустимых решений, улучшая значение целевой функции. Время работы может быть экспоненицальным.
- Метод внутренней точки тоже переходит от точки к точке, улучшая значение целевой функции, но остаётся при это во внутренности многогранника.
- Подобравшись достаточно близко к вершине-решению, быстро находит эту вершину.

Что мы узнали за сегодня?

Что мы узнали за сегодня?

- Симплекс-метод ходит по вершинам многоугольника допустимых решений, улучшая значение целевой функции. Время работы может быть экспоненицальным.
- Метод внутренней точки тоже переходит от точки к точке, улучшая значение целевой функции, но остаётся при это во внутренности многогранника.
- Подобравшись достаточно близко к вершине-решению, быстро находит эту вершину.
- Алгоритм Кармаркара основан на методе внутренней точки и имеет время работы $O(n^4 L)$, где L — длина битовой записи входных данных.

Спасибо за внимание!