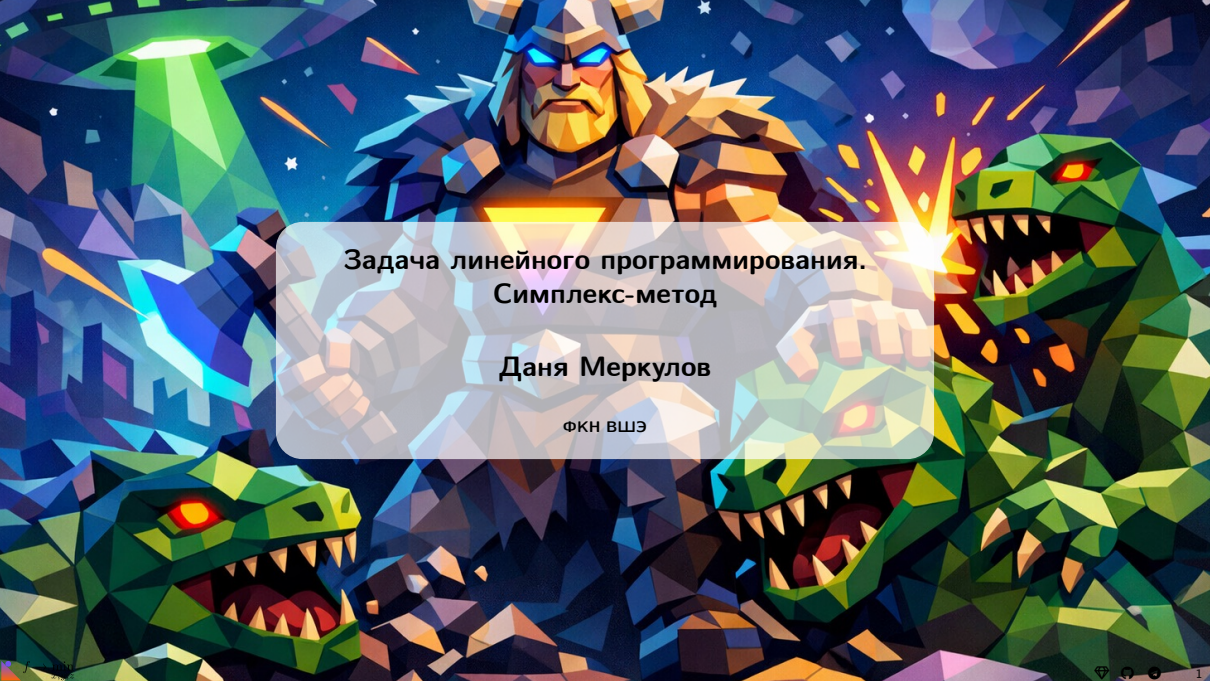


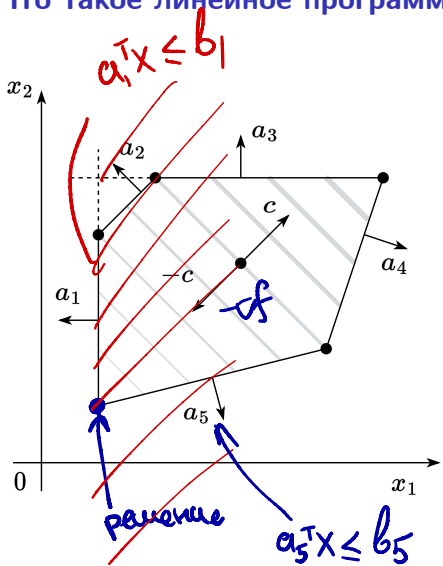
Задача линейного программирования. Симплекс-метод

Даня Меркулов

ФКН ВШЭ

Примеры задач линейного программирования

Что такое линейное программирование?

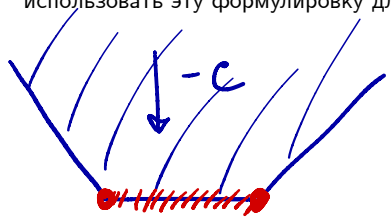


В общем случае все задачи с линейной целевой функцией и линейными функциональными ограничениями можно считать задачами линейного программирования. Однако существует несколько стандартных формулировок.

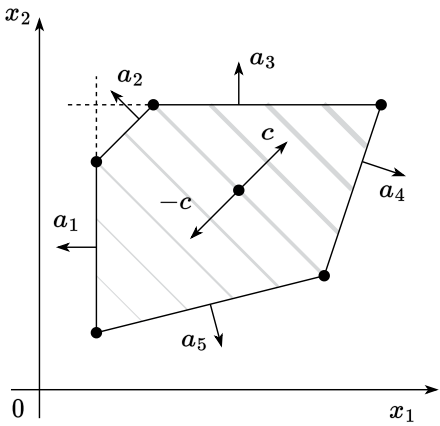
$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \le b$$

(LP.Basic)

для некоторых векторов $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ и матрицы $A \in \mathbb{R}^{m \times n}$, где неравенства — покомпонентные. Мы будем часто использовать эту формулировку для построения интуиции.



Что такое линейное программирование?



В общем случае все задачи с линейной целевой функцией и линейными функциональными ограничениями можно считать задачами линейного программирования. Однако существует несколько стандартных формулировок.

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \leq b \qquad \text{(LP.Basic)}$$

для некоторых векторов $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ и матрицы $A \in \mathbb{R}^{m \times n}$, где неравенства — покомпонентные. Мы будем часто использовать эту формулировку для построения интуиции. Широко используется **стандартная форма** записи задачи линейного программирования. Пусть заданы векторы $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ и матрица $A \in \mathbb{R}^{m \times n}$.

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax = b$$
$$x_i \geq 0,\ i = 1, \dots, n \qquad \text{(LP.Standard)}$$

Пример: задача о диете

Белки	7
Жиры	1
Углеводы	12
Калории	140
Витамин D	500

Количество на 100г

$W \in \mathbb{R}^{n \times p}$

W_j

$c \in \mathbb{R}^p$, цена за 100г

$r \in \mathbb{R}^n$, ограничения

$x \in \mathbb{R}^p$, количество продуктов

$$\min_{x \in \mathbb{R}^p} c^T x$$

$$Wx \succeq r$$

$$x \succeq 0$$

Чижик

X – кол-во продуктов

$x \in \mathbb{R}^p$

стоимость

$C \in \mathbb{R}^p$ – цены

$C^T X \to \min_{x \in \mathbb{R}^p}$

$x \geq 0$

$Wx \geq r_{min}$

$X_{min} \leq X \leq X_{max}$

$X \leq X_{max}$

$x_i \leq b_i$

Пример: задача о диете

Белки	
Жиры	
Углеводы	Количество на 100г
Калории	$W \in \mathbb{R}^{n \times p}$
Витамин D	

$c \in \mathbb{R}^p$, цена за 100г

$r \in \mathbb{R}^n$, ограничения

$x \in \mathbb{R}^p$, количество продуктов

$$\min_{x \in \mathbb{R}^p} c^T x$$

$$Wx \succeq r$$

$$x \succeq 0$$

Представьте, что вам нужно составить план диеты из некоторых продуктов: бананы, пироги, курица, яйца, рыба. Каждый из продуктов имеет свой вектор питательных веществ. Таким образом, все питательные вещества можно представить в виде матрицы W.

Пример: задача о диете

Белки

Жиры

Углеводы

Калории

Витамин D

Количество на 100г

$W \in \mathbb{R}^{n \times p}$

$c \in \mathbb{R}^p$, цена за 100г

$r \in \mathbb{R}^n$, ограничения

$x \in \mathbb{R}^p$, количество продуктов

$$\min_{x \in \mathbb{R}^p} c^T x$$

$$Wx \succeq r$$

$$x \succeq 0$$

Представьте, что вам нужно составить план диеты из некоторых продуктов: бананы, пироги, курица, яйца, рыба. Каждый из продуктов имеет свой вектор питательных веществ. Таким образом, все питательные вещества можно представить в виде матрицы W. Предположим, что у нас есть вектор требований для каждого питательного вещества $r \in \mathbb{R}^n$. Нам нужно найти самую дешёвую диету, которая удовлетворяет всем требованиям:

Пример: задача о диете

Белки

Жиры

Углеводы

Калории

Витамин D

$$\left[\begin{array}{c} \\ \\ \\ \\ \\ \end{array}\right.$$

Количество на 100г

$W \in \mathbb{R}^{n \times p}$

$$\left.\begin{array}{c} \\ \\ \\ \\ \\ \end{array}\right]$$

$c \in \mathbb{R}^p$, цена за 100г

$r \in \mathbb{R}^n$, ограничения

$x \in \mathbb{R}^p$, количество продуктов

$$\min_{x \in \mathbb{R}^p} c^T x$$

$$Wx \succeq r$$

$$x \succeq 0$$

Представьте, что вам нужно составить план диеты из некоторых продуктов: бананы, пироги, курица, яйца, рыба. Каждый из продуктов имеет свой вектор питательных веществ. Таким образом, все питательные вещества можно представить в виде матрицы W. Предположим, что у нас есть вектор требований для каждого питательного вещества $r \in \mathbb{R}^n$. Нам нужно найти самую дешёвую диету, которая удовлетворяет всем требованиям:

$$\begin{aligned} &\min_{x \in \mathbb{R}^p} c^\top x \\ \text{s.t. } &Wx \succeq r \\ &x_i \geq 0,\ i = 1, \dots, p \end{aligned}$$

Open In Colab

Минимизация выпуклой функции как задача линейного программирования

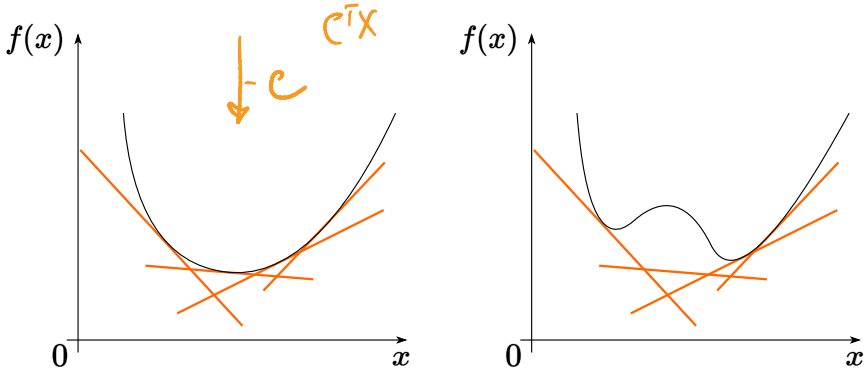


Figure 1: Как задача линейного программирования может помочь с общей задачей выпуклой оптимизации

- Функция выпукла, если она может быть представлена как поточечный максимум линейных функций.

Минимизация выпуклой функции как задача линейного программирования

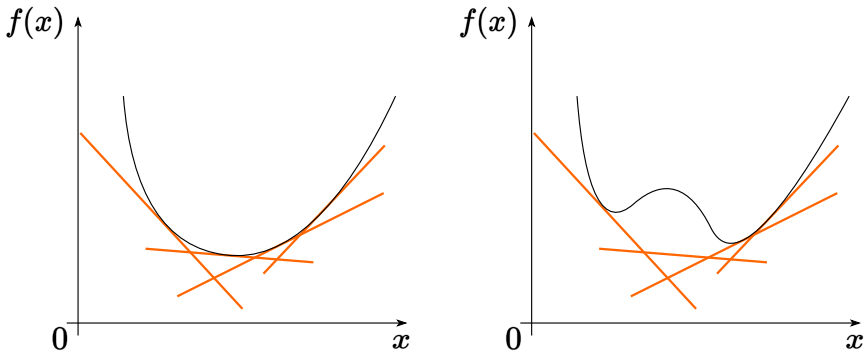


Figure 1: Как задача линейного программирования может помочь с общей задачей выпуклой оптимизации

- Функция выпукла, если она может быть представлена как поточечный максимум линейных функций.
- В пространствах большой размерности аппроксимация может потребовать огромного количества функций.

Минимизация выпуклой функции как задача линейного программирования

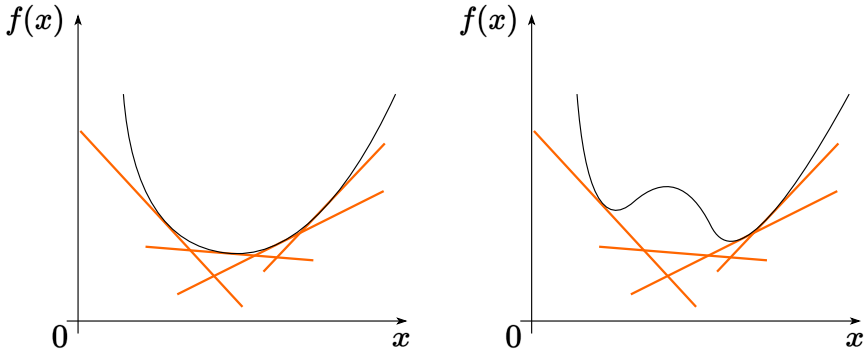


Figure 1: Как задача линейного программирования может помочь с общей задачей выпуклой оптимизации

- Функция выпукла, если она может быть представлена как поточечный максимум линейных функций.
- В пространствах большой размерности аппроксимация может потребовать огромного количества функций.
- Существуют более эффективные солверы для выпуклой оптимизации (не сводящиеся к LP).

Пример: Транспортная задача

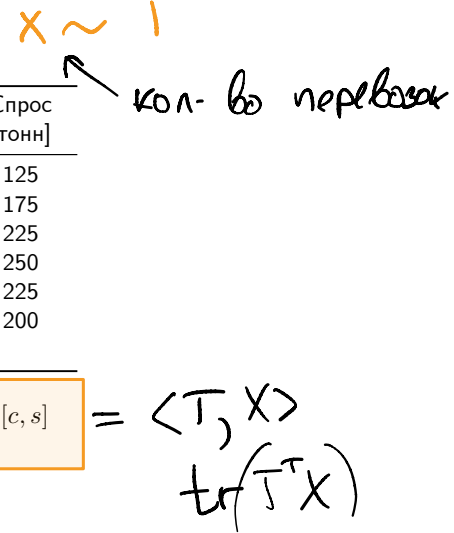
Типичная транспортная задача заключается в распределении товара от производителей к потребителям. Цель состоит в минимизации общих затрат на транспортировку при соблюдении ограничений на количество товара на каждом источнике и удовлетворении требований к спросу на каждом пункте назначения.

Figure 2: Карта Западной Европы. Open In Colab

Пример: Транспортная задача

Пункт назначения / Источник	Арнем [€/тонна]	Гауда [€/тонна]	Спрос [тонн]
Лондон	n/a	2.5	125
Берлин	2.5	n/a	175
Маастрихт	1.6	2.0	225
Амстердам	1.4	1.0	250
Утрехт	0.8	1.0	225
Гаага	1.4	0.8	200
Макс. производство [тонн]	550	700	

Минимизировать: $\text{Стоимость} = \sum_{c \in \text{Пункты назначения}} \sum_{s \in \text{Источники}} T[c,s]x[c,s]$



Пример: Транспортная задача

Пункт назначения / Источник	Арнем [€/тонна]	Гауда [€/тонна]	Спрос [тонн]
Лондон	n/a	2.5	125
Берлин	2.5	n/a	175
Маастрихт	1.6	2.0	225
Амстердам	1.4	1.0	250
Утрехт	0.8	1.0	225
Гаага	1.4	0.8	200
Макс. производство [тонн]	550	700	

Минимизировать: $\text{Стоимость} = \sum_{c \in \text{Пункты назначения}} \sum_{s \in \text{Источники}} T[c,s]x[c,s]$

$$\sum_{c \in \text{Пункты назначения}} x[c,s] \leq \text{Поставка}[s] \qquad \forall s \in \text{Источники}$$

Пример: Транспортная задача

Пункт назначения / Источник	Арнем [€/тонна]	Гауда [€/тонна]	Спрос [тонн]
Лондон	n/a	2.5	125
Берлин	2.5	n/a	175
Маастрихт	1.6	2.0	225
Амстердам	1.4	1.0	250
Утрехт	0.8	1.0	225
Гаага	1.4	0.8	200
Макс. производство [тонн]	550	700	

Минимизировать: Стоимость $= \sum_{c \in \text{Пункты назначения}} \sum_{s \in \text{Источники}} T[c,s]x[c,s]$

$$\sum_{c \in \text{Пункты назначения}} x[c,s] \leq \text{Поставка}[s] \qquad \forall s \in \text{Источники}$$

$$\sum_{s \in \text{Источники}} x[c,s] = \text{Спрос}[c] \qquad \forall c \in \text{Пункты назначения}$$

Задачу можно представить в виде следующего графа:

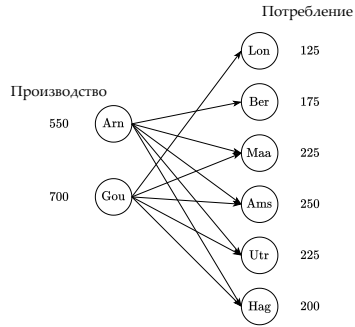


Figure 3: Граф, связанный с задачей

Как получить задачу линейного программирования?

Основные преобразования

- Максимум-минимум

$$\min_{x \in \mathbb{R}^n} c^\top x \quad \text{s.t. } Ax \leq b \quad \leftrightarrow \quad \max_{x \in \mathbb{R}^n} -c^\top x \quad \text{s.t. } Ax \leq b$$

Основные преобразования

- Максимум-минимум

$$\min_{x \in \mathbb{R}^n} c^\top x \quad \text{s.t. } Ax \leq b \qquad \leftrightarrow \qquad \max_{x \in \mathbb{R}^n} -c^\top x \quad \text{s.t. } Ax \leq b$$

- Равенство к неравенству

$$Ax = b \leftrightarrow \begin{cases} Ax \leq b \\ Ax \geq b \end{cases}$$

$$Ax \leq b$$
$$-Ax \leq -b$$

Основные преобразования

- Максимум-минимум

$$\min_{x \in \mathbb{R}^n} c^\top x \quad \text{s.t. } Ax \leq b \qquad \leftrightarrow \qquad \max_{x \in \mathbb{R}^n} -c^\top x \quad \text{s.t. } Ax \leq b$$

- Равенство к неравенству

$$Ax = b \leftrightarrow \begin{cases} Ax \leq b \\ Ax \geq b \end{cases}$$

- Неравенство к равенству, увеличивая размерность задачи на m.

$$Ax \leq b \leftrightarrow \begin{cases} Ax + z = b \\ z \geq 0 \end{cases}$$

$z = b - Ax \geqslant 0$

$Ax + z = b$

$z \geqslant 0 \quad b \geqslant Ax$

Основные преобразования

- Максимум-минимум

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} c^\top x \\ \text{s.t. } &Ax \le b \end{aligned} \quad \leftrightarrow \quad \begin{aligned} &\max_{x\in\mathbb{R}^n} -c^\top x \\ \text{s.t. } &Ax \le b \end{aligned}$$

- Равенство к неравенству

$$Ax = b \leftrightarrow \begin{cases} Ax \le b \\ Ax \ge b \end{cases}$$

- Неравенство к равенству, увеличивая размерность задачи на m.

$$Ax \le b \leftrightarrow \begin{cases} Ax + z = b \\ z \ge 0 \end{cases}$$

- Неотрицательные переменные

$$x \leftrightarrow \begin{cases} x = x_+ - x_- \\ x_+ \ge 0 \\ x_- \ge 0 \end{cases}$$

$$X = \begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix} = X_+ - X_- \qquad X_+ \begin{pmatrix} 1 \\ 0 \\ 3 \end{pmatrix} \quad X_- \begin{pmatrix} 0 \\ 2 \\ 0 \end{pmatrix}$$

Пример: задача аппроксимации Чебышева

$$p = Ax - b$$

$$p_i = a_i^T x - b_i$$

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_\infty \leftrightarrow \min_{x \in \mathbb{R}^n} \max_i |a_i^T x - b_i|$$

Можно записать эквивалентную задачу линейного программирования с заменой максимальной координаты вектора:

Пусть $\max_i |p_i| = t$

$\forall j$: $-t \leqslant p_j \leqslant t$

LP

$$\min_{x,t} t$$

$$-t \leqslant a_j^T x - b_j \leqslant t$$

Пример: задача аппроксимации Чебышева

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_\infty \leftrightarrow \min_{x \in \mathbb{R}^n} \max_i |a_i^T x - b_i|$$

Можно записать эквивалентную задачу линейного программирования с заменой максимальной координаты вектора:

$$\begin{aligned} &\min_{t \in \mathbb{R}, x \in \mathbb{R}^n} t \\ \text{s.t. } &a_i^T x - b_i \leq t, \ i = 1, \dots, m \\ &-a_i^T x + b_i \leq t, \ i = 1, \dots, m \end{aligned}$$

Пример: задача ℓ_1 аппроксимации

$$|a_i^T x - b_i| = t_i$$

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_1 \leftrightarrow \min_{x \in \mathbb{R}^n} \sum_{i=1}^{m} |a_i^T x - b_i| = \min_{x,t} \sum_{i=1}^{m} t_i =$$

Можно записать эквивалентную задачу линейного программирования с заменой суммы координат вектора:

$$= \min 1^T t$$

$$-t_i \leq a_i^T x - b_i \leq t_i$$

Пример: задача ℓ_1 аппроксимации

$$\min_{x \in \mathbb{R}^n} \|Ax - b\|_1 \leftrightarrow \min_{x \in \mathbb{R}^n} \sum_{i=1}^{m} |a_i^T x - b_i|$$

Можно записать эквивалентную задачу линейного программирования с заменой суммы координат вектора:

$$|a_i^T x - b_i| = t_i$$

$$\min_{t \in \mathbb{R}^m, x \in \mathbb{R}^n} \mathbf{1}^T t$$

$$\text{s.t. } a_i^T x - b_i \leq t_i,\ i = 1, \dots, m$$

$$-a_i^T x + b_i \leq t_i,\ i = 1, \dots, m$$

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат A (Добрый кола)	10.6	1.25
Концентрат B (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смесь с минимальной стоимостью, которая удовлетворит заказ.

Целевая функция

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат A (Добрый кола)	10.6	1.25
Концентрат B (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смесь с минимальной стоимостью, которая удовлетворит заказ.

Целевая функция

Минимизировать стоимость:

$$\text{Cost} = \sum_{c \in C} x_c \cdot P_c$$

где x_c — объём используемого компонента c, и P_c — его цена.

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Ограничение на объём

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат А (Добрый кола)	10.6	1.25
Концентрат В (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смесь с минимальной стоимостью, которая удовлетворит заказ.

Целевая функция

Минимизировать стоимость:

$$\text{Cost} = \sum_{c \in C} x_c \cdot P_c$$

где x_c — объём используемого компонента c, и P_c — его цена.

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат А (Добрый кола)	10.6	1.25
Концентрат В (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смесь с минимальной стоимостью, которая удовлетворит заказ.

Целевая функция

Минимизировать стоимость:

$$\text{Cost} = \sum_{c \in C} x_c \cdot P_c$$

где x_c — объём используемого компонента c, и P_c — его цена.

Ограничение на объём

Убедитесь, что общий объём V:

$$V = \sum_{c \in C} x_c$$

Ограничение на состав

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат A (Добрый кола)	10.6	1.25
Концентрат B (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смесь с минимальной стоимостью, которая удовлетворит заказ.

Целевая функция

Минимизировать стоимость:

$$\text{Cost} = \sum_{c \in C} x_c \cdot P_c$$

где x_c — объём используемого компонента c, и P_c — его цена.

Ограничение на объём

Убедитесь, что общий объём V:

$$V = \sum_{c \in C} x_c$$

Ограничение на состав

Убедитесь, что содержание сахара — 4%:

$$\bar{A} = \frac{\sum_{c \in C} x_c A_c}{\sum_{c \in C} x_c}$$

[1] Source

Задача смешивания: от нелинейных ограничений к ЛП [1]

Производственное предприятие получает заказ на 100 литров раствора с определённой концентрацией (например, 4% сахарного раствора). На складе есть:

Компонент	Сахар (%)	Стоимость ($/л)
Концентрат А (Добрый кола)	10.6	1.25
Концентрат В (Север кола)	4.5	1.02
Вода (Псыж)	0.0	0.62

Цель: Найти смесь с минимальной стоимостью, которая удовлетворит заказ.

Целевая функция

Минимизировать стоимость:

$$\text{Cost} = \sum_{c \in C} x_c \cdot P_c$$

где x_c — объём используемого компонента c, и P_c — его цена.

Ограничение на объём

Убедитесь, что общий объём V:

$$V = \sum_{c \in C} x_c$$

Ограничение на состав

Убедитесь, что содержание сахара — 4%:

$$\bar{A} = \frac{\sum_{c \in C} x_c A_c}{\sum_{c \in C} x_c}$$

Линеаризованная версия:

$$0 = \sum_{c \in C} x_c (A_c - \bar{A})$$

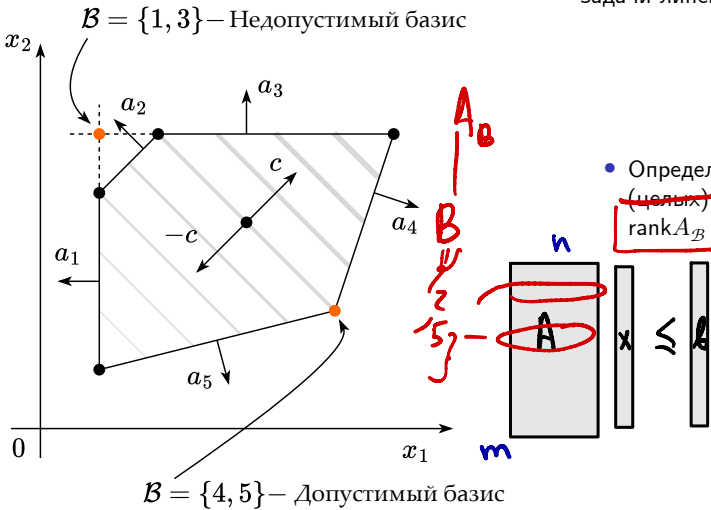
Это можно решить с помощью линейного программирования.

Код

[1] Source

Симплекс-метод

Геометрия симплекс-метода



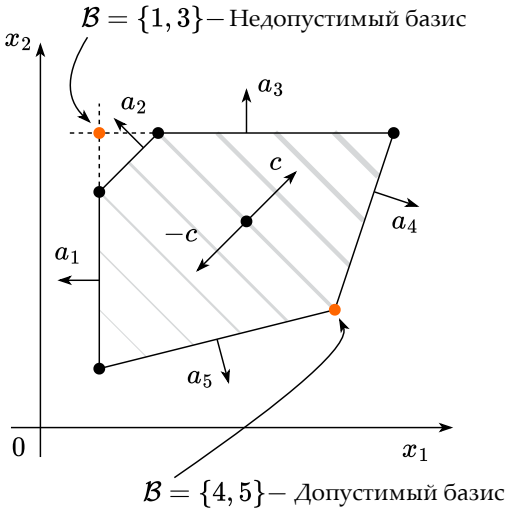
Рассмотрим следующую простую формулировку задачи линейного программирования:

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \le b$$

(LP.Inequality)

- Определение: **базис** $\mathcal{B}$ — это подмножество n (целых) чисел между 1 и m, такое что $\text{rank} A_{\mathcal{B}} = n$.

Геометрия симплекс-метода

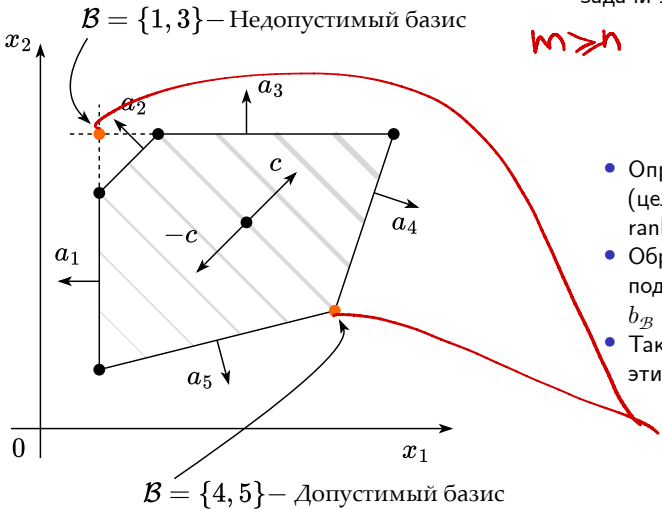


Рассмотрим следующую простую формулировку задачи линейного программирования:

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \le b \qquad \text{(LP.Inequality)}$$

- Определение: **базис** $\mathcal{B}$ — это подмножество n (целых) чисел между 1 и m, такое что $\text{rank} A_{\mathcal{B}} = n$.
- Обратите внимание, что мы можем связать подматрицу $A_{\mathcal{B}}$ и соответствующую правую часть $b_{\mathcal{B}}$ с базисом $\mathcal{B}$.

Геометрия симплекс-метода



Рассмотрим следующую простую формулировку задачи линейного программирования:

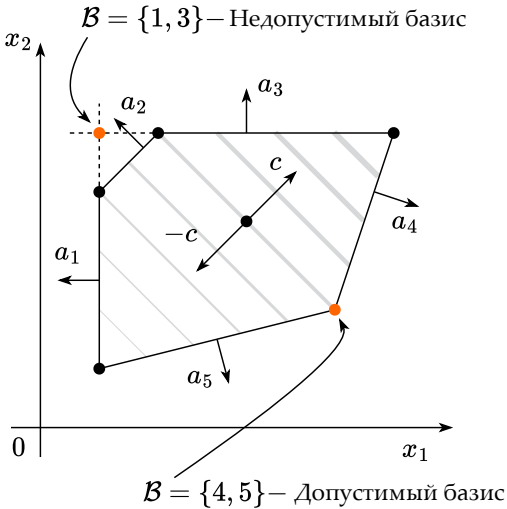
$m \gg n$

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \leq b \qquad \text{(LP.Inequality)}$$

- Определение: **базис** $\mathcal{B}$ — это подмножество n (целых) чисел между 1 и m, такое что $\text{rank} A_{\mathcal{B}} = n$.
- Обратите внимание, что мы можем связать подматрицу $A_{\mathcal{B}}$ и соответствующую правую часть $b_{\mathcal{B}}$ с базисом $\mathcal{B}$.
- Также мы можем получить точку пересечения всех этих гиперплоскостей из базиса: $x_{\mathcal{B}} = A_{\mathcal{B}}^{-1} b_{\mathcal{B}}$.

$$x_B = A_B^{-1} b_B$$

Геометрия симплекс-метода

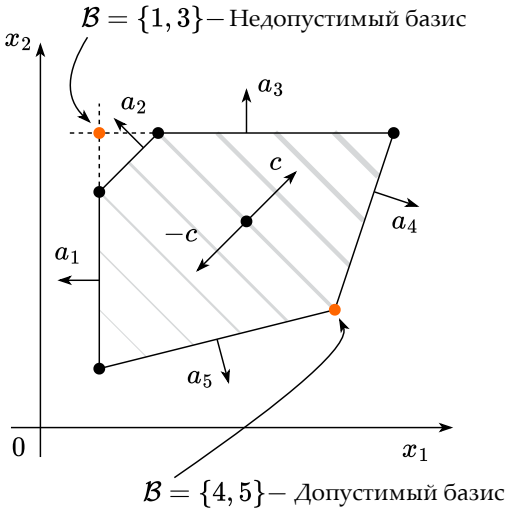


Рассмотрим следующую простую формулировку задачи линейного программирования:

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \le b \qquad \text{(LP.Inequality)}$$

- Определение: **базис** $\mathcal{B}$ — это подмножество n (целых) чисел между 1 и m, такое что $\text{rank} A_{\mathcal{B}} = n$.
- Обратите внимание, что мы можем связать подматрицу $A_{\mathcal{B}}$ и соответствующую правую часть $b_{\mathcal{B}}$ с базисом $\mathcal{B}$.
- Также мы можем получить точку пересечения всех этих гиперплоскостей из базиса: $x_{\mathcal{B}} = A_{\mathcal{B}}^{-1} b_{\mathcal{B}}$.
- Если $Ax_{\mathcal{B}} \le b$, то базис $\mathcal{B}$ является **допустимым**.

Геометрия симплекс-метода

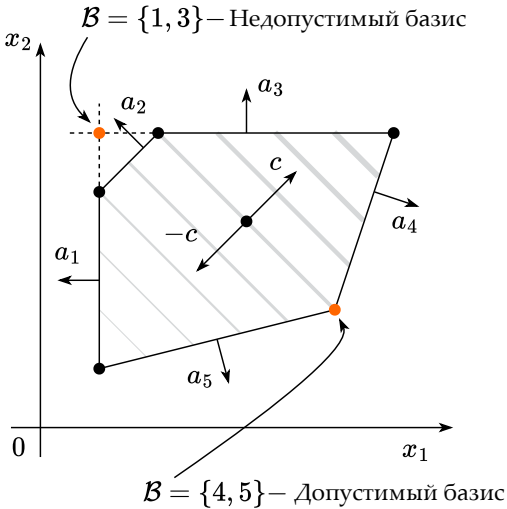


Рассмотрим следующую простую формулировку задачи линейного программирования:

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \le b \qquad \text{(LP.Inequality)}$$

- Определение: **базис** $\mathcal{B}$ — это подмножество n (целых) чисел между 1 и m, такое что $\text{rank} A_{\mathcal{B}} = n$.
- Обратите внимание, что мы можем связать подматрицу $A_{\mathcal{B}}$ и соответствующую правую часть $b_{\mathcal{B}}$ с базисом $\mathcal{B}$.
- Также мы можем получить точку пересечения всех этих гиперплоскостей из базиса: $x_{\mathcal{B}} = A_{\mathcal{B}}^{-1} b_{\mathcal{B}}$.
- Если $Ax_{\mathcal{B}} \le b$, то базис $\mathcal{B}$ является **допустимым**.
- Базис $\mathcal{B}$ оптимален, если $x_{\mathcal{B}}$ является решением задачи LP.Inequality.

Геометрия симплекс-метода



Рассмотрим следующую простую формулировку задачи линейного программирования:

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \leq b \qquad \text{(LP.Inequality)}$$

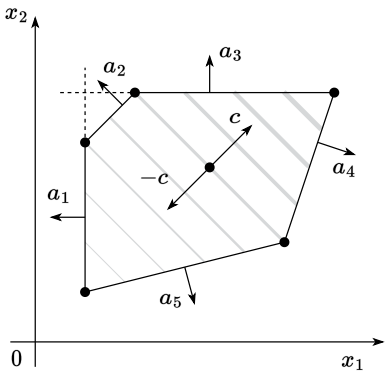
- Определение: **базис** $\mathcal{B}$ — это подмножество n (целых) чисел между 1 и m, такое что $\text{rank} A_{\mathcal{B}} = n$.
- Обратите внимание, что мы можем связать подматрицу $A_{\mathcal{B}}$ и соответствующую правую часть $b_{\mathcal{B}}$ с базисом $\mathcal{B}$.
- Также мы можем получить точку пересечения всех этих гиперплоскостей из базиса: $x_{\mathcal{B}} = A_{\mathcal{B}}^{-1} b_{\mathcal{B}}$.
- Если $Ax_{\mathcal{B}} \leq b$, то базис $\mathcal{B}$ является **допустимым**.
- Базис $\mathcal{B}$ оптимален, если $x_{\mathcal{B}}$ является решением задачи LP.Inequality.
- $x_{\mathcal{B}}$ называют **базисной точкой** или базисным решением (иногда её тоже называют **базисом**).

Если решение задачи линейного программирования существует, то оно лежит в вершине

$c^T x \to \min$

$Ax = b$

$x \geq 0$

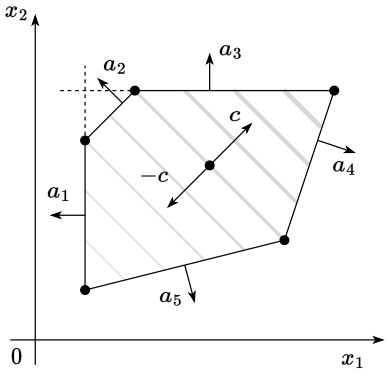


Theorem

1. Если задача линейного программирования в стандартной форме имеет непустое бюджетное множество, то существует по крайней мере одна допустимая базисная точка.

Верхнеуровневая идея симплекс-метода:

Если решение задачи линейного программирования существует, то оно лежит в вершине

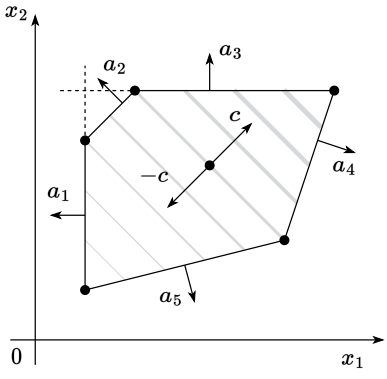


Theorem

1. Если задача линейного программирования в стандартной форме имеет непустое бюджетное множество, то существует по крайней мере одна допустимая базисная точка.
2. Если задача линейного программирования в стандартной форме имеет решения, то по крайней мере одно из таких решений является оптимальной базисной точкой.

Верхнеуровневая идея симплекс-метода:

Если решение задачи линейного программирования существует, то оно лежит в вершине



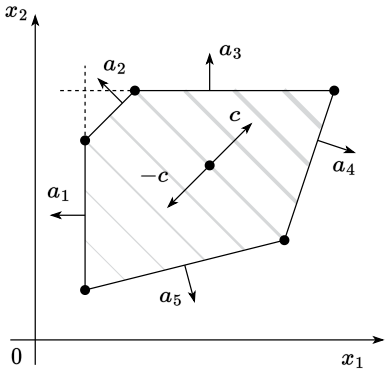
Theorem

1. Если задача линейного программирования в стандартной форме имеет непустое бюджетное множество, то существует по крайней мере одна допустимая базисная точка.
2. Если задача линейного программирования в стандартной форме имеет решения, то по крайней мере одно из таких решений является оптимальной базисной точкой.
3. Если задача линейного программирования в стандартной форме допустима и ограничена, то она имеет оптимальное решение.

бюджетное мн-во не пусто

Верхнеуровневая идея симплекс-метода:

Если решение задачи линейного программирования существует, то оно лежит в вершине

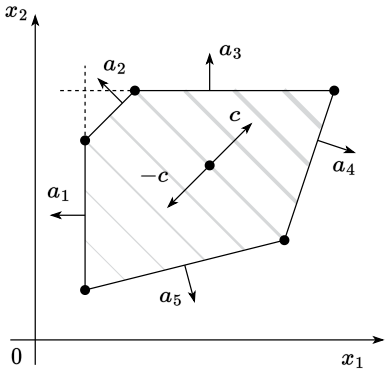


Theorem

1. Если задача линейного программирования в стандартной форме имеет непустое бюджетное множество, то существует по крайней мере одна допустимая базисная точка.
2. Если задача линейного программирования в стандартной форме имеет решения, то по крайней мере одно из таких решений является оптимальной базисной точкой.
3. Если задача линейного программирования в стандартной форме допустима и ограничена, то она имеет оптимальное решение.

Верхнеуровневая идея симплекс-метода:

Если решение задачи линейного программирования существует, то оно лежит в вершине



Theorem

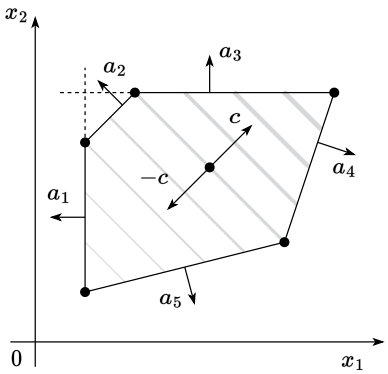
1. Если задача линейного программирования в стандартной форме имеет непустое бюджетное множество, то существует по крайней мере одна допустимая базисная точка.
2. Если задача линейного программирования в стандартной форме имеет решения, то по крайней мере одно из таких решений является оптимальной базисной точкой.
3. Если задача линейного программирования в стандартной форме допустима и ограничена, то она имеет оптимальное решение.

Для доказательства см. теорему 13.2 в Numerical Optimization by Jorge Nocedal and Stephen J. Wright

Верхнеуровневая идея симплекс-метода:

- Убедитесь, что вы находитесь в вершине.

Если решение задачи линейного программирования существует, то оно лежит в вершине



Theorem

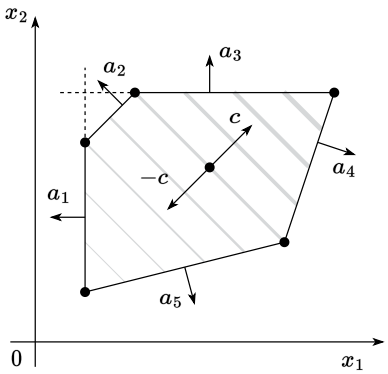
1. Если задача линейного программирования в стандартной форме имеет непустое бюджетное множество, то существует по крайней мере одна допустимая базисная точка.
2. Если задача линейного программирования в стандартной форме имеет решения, то по крайней мере одно из таких решений является оптимальной базисной точкой.
3. Если задача линейного программирования в стандартной форме допустима и ограничена, то она имеет оптимальное решение.

Для доказательства см. теорему 13.2 в Numerical Optimization by Jorge Nocedal and Stephen J. Wright

Верхнеуровневая идея симплекс-метода:

- Убедитесь, что вы находитесь в вершине.
- Проверьте оптимальность.

Если решение задачи линейного программирования существует, то оно лежит в вершине



Theorem

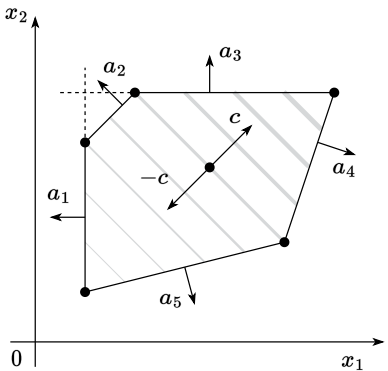
1. Если задача линейного программирования в стандартной форме имеет непустое бюджетное множество, то существует по крайней мере одна допустимая базисная точка.
2. Если задача линейного программирования в стандартной форме имеет решения, то по крайней мере одно из таких решений является оптимальной базисной точкой.
3. Если задача линейного программирования в стандартной форме допустима и ограничена, то она имеет оптимальное решение.

Для доказательства см. теорему 13.2 в Numerical Optimization by Jorge Nocedal and Stephen J. Wright

Верхнеуровневая идея симплекс-метода:

- Убедитесь, что вы находитесь в вершине.
- Проверьте оптимальность.
- Если необходимо, перейдите к другой вершине (измените базис).

Если решение задачи линейного программирования существует, то оно лежит в вершине



Theorem

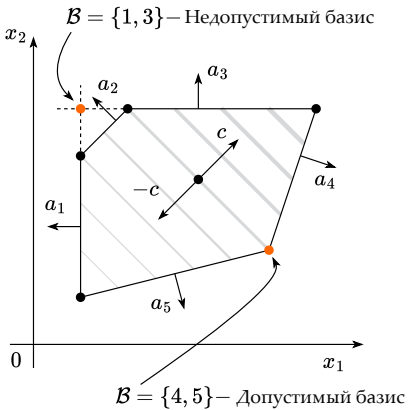
1. Если задача линейного программирования в стандартной форме имеет непустое бюджетное множество, то существует по крайней мере одна допустимая базисная точка.
2. Если задача линейного программирования в стандартной форме имеет решения, то по крайней мере одно из таких решений является оптимальной базисной точкой.
3. Если задача линейного программирования в стандартной форме допустима и ограничена, то она имеет оптимальное решение.

Для доказательства см. теорему 13.2 в Numerical Optimization by Jorge Nocedal and Stephen J. Wright

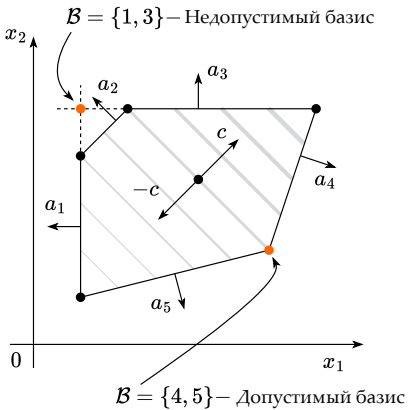
Верхнеуровневая идея симплекс-метода:

- Убедитесь, что вы находитесь в вершине.
- Проверьте оптимальность.
- Если необходимо, перейдите к другой вершине (измените базис).
- Повторяйте, пока не сойдётесь.

Оптимальный базис



Оптимальный базис

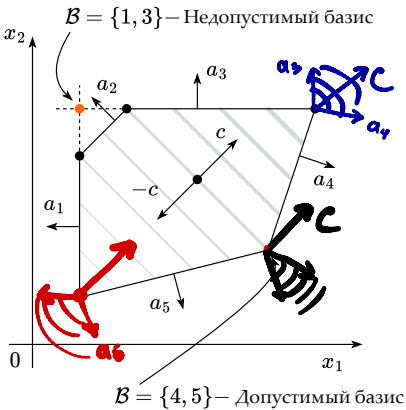


B $\quad A_B, b_B \rightarrow x_B = A_B^{-1} b_B$

Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

Оптимальный базис



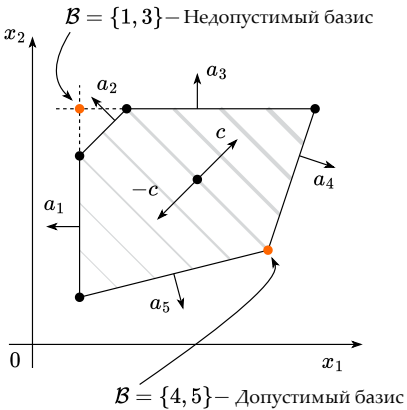
Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \boxed{\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}}$$

Theorem

Если все компоненты $\lambda_{\mathcal{B}}$ неположительны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Оптимальный базис



Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

Theorem

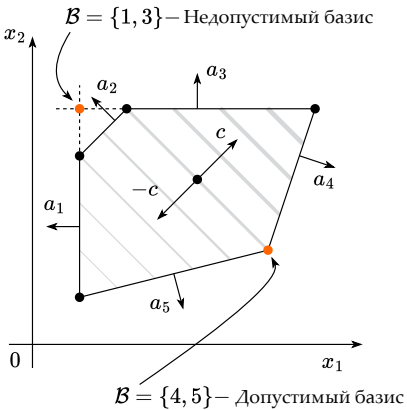
Если все компоненты $\lambda_{\mathcal{B}}$ неположительны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Доказательство Предположим противное, то есть $\lambda_{\mathcal{B}} \leq 0$ и $\mathcal{B}$ допустим, но не оптимален.

$$\exists x^* : Ax^* \leq b, c^T x^* < c^T x_{\mathcal{B}}$$

$X_{\mathcal{B}}$ не оптимальна

Оптимальный базис



Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

Theorem

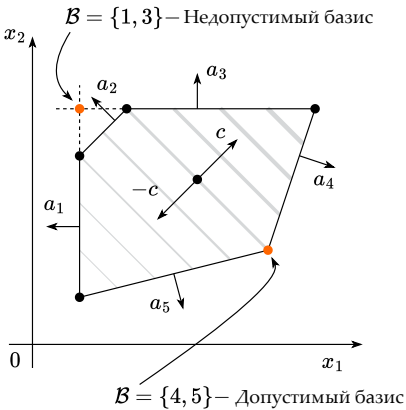
Если все компоненты $\lambda_{\mathcal{B}}$ неположительны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Доказательство Предположим противное, то есть $\lambda_{\mathcal{B}} \leq 0$ и $\mathcal{B}$ допустим, но не оптимален.

$$\exists x^* : Ax^* \leq b, c^T x^* < c^T x_{\mathcal{B}}$$

$$A_{\mathcal{B}} x^* \leq b_{\mathcal{B}}$$

Оптимальный базис



Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

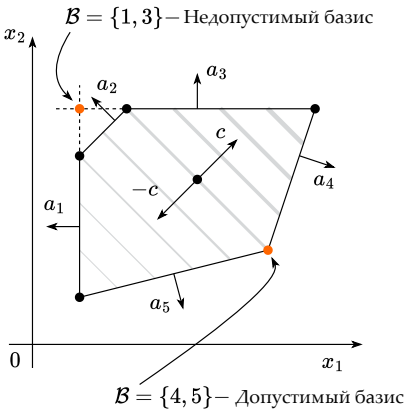
Theorem

Если все компоненты $\lambda_{\mathcal{B}}$ неположительны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Доказательство Предположим противное, то есть $\lambda_{\mathcal{B}} \leq 0$ и $\mathcal{B}$ допустим, но не оптимален.

$$\exists x^* : Ax^* \leq b, c^T x^* < c^T x_{\mathcal{B}}$$
$$A_{\mathcal{B}} x^* \leq b_{\mathcal{B}} \mid \lambda_{\mathcal{B}}^T \cdot \leq 0$$

Оптимальный базис



Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

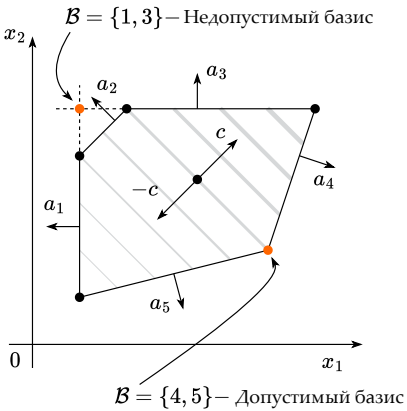
Theorem

Если все компоненты $\lambda_{\mathcal{B}}$ неположительны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Доказательство Предположим противное, то есть $\lambda_{\mathcal{B}} \leq 0$ и $\mathcal{B}$ допустим, но не оптимален.

$$\exists x^* : Ax^* \leq b, c^T x^* < c^T x_{\mathcal{B}}$$
$$A_{\mathcal{B}} x^* \leq b_{\mathcal{B}} \mid \lambda_{\mathcal{B}}^T \cdot \leq 0$$
$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} x^* \geq \lambda_{\mathcal{B}}^T b_{\mathcal{B}}$$

Оптимальный базис



Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

Theorem

Если все компоненты $\lambda_{\mathcal{B}}$ неположительны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Доказательство Предположим противное, то есть $\lambda_{\mathcal{B}} \leq 0$ и $\mathcal{B}$ допустим, но не оптимален.

$$\exists x^* : Ax^* \leq b, c^T x^* < c^T x_{\mathcal{B}}$$

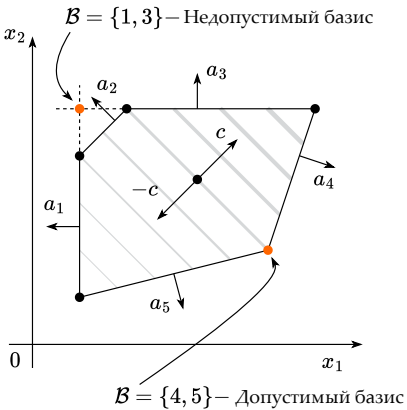
$$A_{\mathcal{B}} x^* \leq b_{\mathcal{B}} \mid \lambda_{\mathcal{B}}^T \cdot \leq 0$$

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} x^* \geq \lambda_{\mathcal{B}}^T b_{\mathcal{B}}$$

$$c^T x^* \geq \lambda_{\mathcal{B}}^T A_{\mathcal{B}} x_{\mathcal{B}}$$

c

Оптимальный базис



Поскольку у нас есть базис, мы можем разложить наш целевой вектор c в этом базисе и найти скалярные коэффициенты $\lambda_{\mathcal{B}}$:

$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} = c^T \leftrightarrow \lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$$

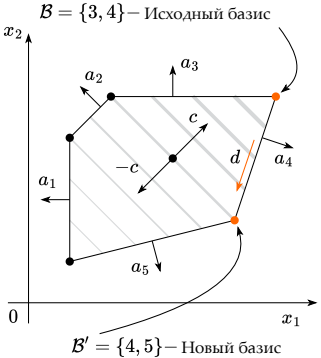
Theorem

Если все компоненты $\lambda_{\mathcal{B}}$ неположительны и $\mathcal{B}$ допустим, то $\mathcal{B}$ оптимален.

Доказательство Предположим противное, то есть $\lambda_{\mathcal{B}} \leq 0$ и $\mathcal{B}$ допустим, но не оптимален.

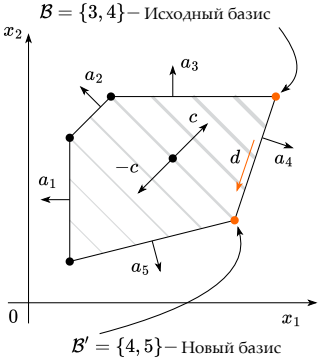
$$\exists x^* : Ax^* \leq b, \boxed{c^T x^* < c^T x_{\mathcal{B}}}$$
$$A_{\mathcal{B}} x^* \leq b_{\mathcal{B}} \mid \lambda_{\mathcal{B}}^T \cdot \leq 0$$
$$\lambda_{\mathcal{B}}^T A_{\mathcal{B}} x^* \geq \lambda_{\mathcal{B}}^T b_{\mathcal{B}}$$
$$c^T x^* \geq \lambda_{\mathcal{B}}^T A_{\mathcal{B}} x_{\mathcal{B}}$$
$$\boxed{c^T x^* \geq c^T x_{\mathcal{B}}}$$

Изменение базиса



Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

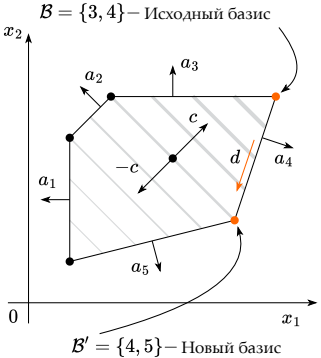
Изменение базиса



- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

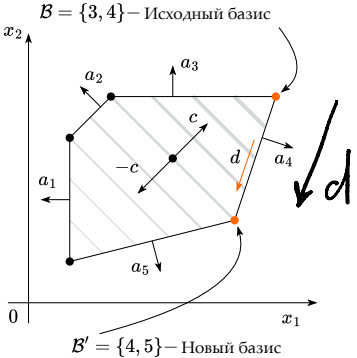
Изменение базиса



- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый.

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса

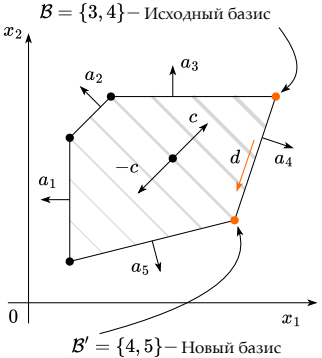


- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

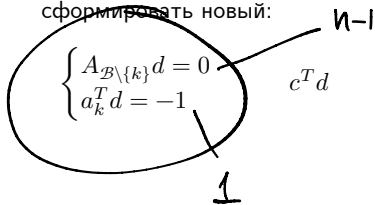
$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} < 0$$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса

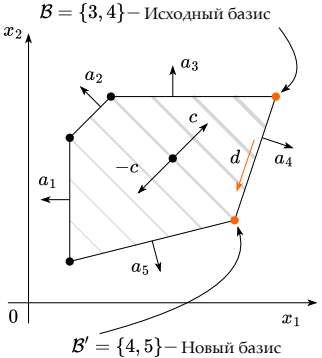


- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:



Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса

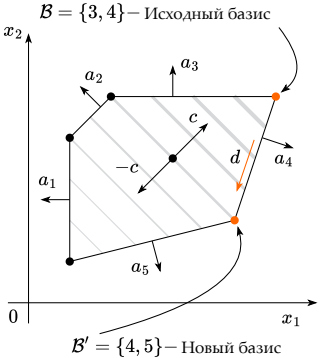


- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad \underbrace{c^T d} = \underbrace{\lambda_{\mathcal{B}}^T A_{\mathcal{B}} d}$$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса

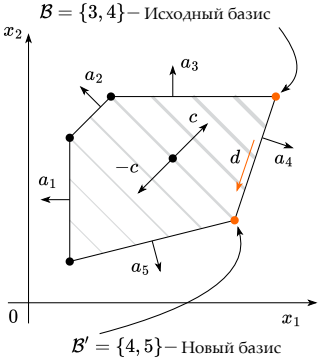


- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda_{\mathcal{B}}^T A_{\mathcal{B}} d = \sum_{i=1}^{n} \lambda_{\mathcal{B}}^i (A_{\mathcal{B}} d)^i$$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса



- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda_{\mathcal{B}}^T A_{\mathcal{B}} d = \sum_{i=1}^{n} \lambda_{\mathcal{B}}^i (A_{\mathcal{B}} d)^i = -\lambda_{\mathcal{B}}^k < 0$$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса

- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

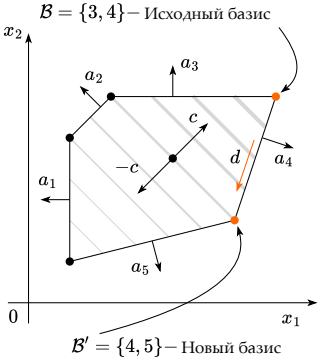
$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda_{\mathcal{B}}^T A_{\mathcal{B}} d = \sum_{i=1}^{n} \lambda_{\mathcal{B}}^i (A_{\mathcal{B}} d)^i = -\lambda_{\mathcal{B}}^k < 0$$

- Для всех $j \notin \mathcal{B}$ рассчитаем размер шага проекции:

$$\mu_j = \frac{b_j - a_j^T x_{\mathcal{B}}}{a_j^T d}$$

Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

Изменение базиса



Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda_{\mathcal{B}}^T A_{\mathcal{B}} d = \sum_{i=1}^{n} \lambda_{\mathcal{B}}^i (A_{\mathcal{B}} d)^i = -\lambda_{\mathcal{B}}^k < 0$$

- Для всех $j \notin \mathcal{B}$ рассчитаем размер шага проекции:

$$\mu_j = \frac{b_j - a_j^T x_{\mathcal{B}}}{a_j^T d}$$

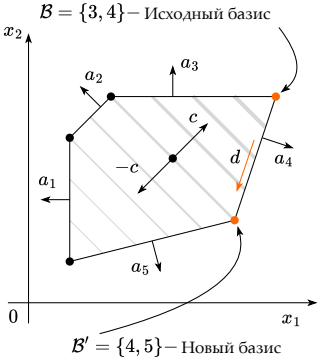
- Определим новую вершину, которую мы добавим в новый базис:

$$t = \arg\min_j \{\mu_j \mid \mu_j > 0\}$$

$$\mathcal{B}' = \mathcal{B}\setminus\{k\} \cup \{t\}$$

$$x_{\mathcal{B}'} = x_{\mathcal{B}} + \mu_t d = A_{\mathcal{B}'}^{-1} b_{\mathcal{B}'}$$

Изменение базиса



Предположим, что некоторые из коэффициентов $\lambda_{\mathcal{B}}$ положительны. В этом случае необходимо осуществить переход по ребру многогранника к новой вершине, то есть произвести замену базиса.

- Предположим, что у нас есть базис $\mathcal{B}$: $\lambda_{\mathcal{B}}^T = c^T A_{\mathcal{B}}^{-1}$
- Предположим, что $\lambda_{\mathcal{B}}^k > 0$. Мы хотим удалить k из базиса и сформировать новый:

$$\begin{cases} A_{\mathcal{B}\setminus\{k\}} d = 0 \\ a_k^T d = -1 \end{cases} \qquad c^T d = \lambda_{\mathcal{B}}^T A_{\mathcal{B}} d = \sum_{i=1}^{n} \lambda_{\mathcal{B}}^i (A_{\mathcal{B}} d)^i = -\lambda_{\mathcal{B}}^k < 0$$

- Для всех $j \notin \mathcal{B}$ рассчитаем размер шага проекции:

$$\mu_j = \frac{b_j - a_j^T x_{\mathcal{B}}}{a_j^T d}$$

- Определим новую вершину, которую мы добавим в новый базис:

$$t = \arg\min_j \{\mu_j \mid \mu_j > 0\}$$

$$\mathcal{B}' = \mathcal{B}\setminus\{k\} \cup \{t\}$$

$$x_{\mathcal{B}'} = x_{\mathcal{B}} + \mu_t d = A_{\mathcal{B}'}^{-1} b_{\mathcal{B}'}$$

- Обратите внимание, что изменение базиса приводит к уменьшению целевой функции: $c^T x_{\mathcal{B}'} = c^T (x_{\mathcal{B}} + \mu_t d) = c^T x_{\mathcal{B}} + \mu_t c^T d$

Поиск начального допустимого базиса

Нам нужно решить следующую задачу:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} c^\top x \\ &\text{s.t. } Ax \leq b \end{aligned} \tag{1}$$

Предложенный алгоритм требует начального допустимого базиса.

Поиск начального допустимого базиса

Нам нужно решить следующую задачу:

$$\min_{x \in \mathbb{R}^n} c^\top x \quad \text{s.t. } Ax \leq b \tag{1}$$

Предложенный алгоритм требует начального допустимого базиса.

Начнём с переформулировки задачи:

$$\min_{y \in \mathbb{R}^n, z \in \mathbb{R}^n} c^\top (y - z) \quad \text{s.t. } Ay - Az \leq b, \; y \geq 0, z \geq 0 \tag{2}$$

Поиск начального допустимого базиса

Нам нужно решить следующую задачу:

$$\min_{x \in \mathbb{R}^n} c^\top x$$
$$\text{s.t. } Ax \leq b \tag{1}$$

Предложенный алгоритм требует начального допустимого базиса.

Начнём с переформулировки задачи:

$$\min_{y \in \mathbb{R}^n, z \in \mathbb{R}^n} c^\top (y - z)$$
$$\text{s.t. } Ay - Az \leq b \tag{2}$$
$$y \geq 0, z \geq 0$$

Зная решение задачи (2), можно восстановить решение задачи (1), и наоборот.

$$x = y - z \qquad \Leftrightarrow \qquad y_i = \max(x_i, 0), \quad z_i = \max(-x_i, 0)$$

Теперь мы попытаемся сформулировать новую задачу линейного программирования, решение которой будет допустимой базисной точкой для Задачи 2. Это означает, что мы сначала запускаем симплекс-метод для задачи Phase-1, а затем запускаем задачу Phase-2 с известным начальным решением. Обратите внимание, что допустимое базисное решение для Phase-1 должно быть легко вычислимо.

Поиск начального допустимого базиса

$$\min_{y \in \mathbb{R}^n, z \in \mathbb{R}^n} c^\top (y - z)$$

s.t. $Ay - Az \leq b$ (Фаза-2 (главная задача ЛП))

$y \geq 0, z \geq 0$

Поиск начального допустимого базиса

$$\min_{y\in\mathbb{R}^n, z\in\mathbb{R}^n} c^\top (y - z)$$

s.t. $Ay - Az \le b$ (Фаза-2 (главная задача ЛП))

$y \ge 0, z \ge 0$

$$\min_{\xi\in\mathbb{R}^m, y\in\mathbb{R}^n, z\in\mathbb{R}^n} \sum_{i=1}^{m} \xi_i$$

s.t. $Ay - Az \le b + \xi$ (Фаза-1)

$y \ge 0, z \ge 0, \xi \ge 0$

Поиск начального допустимого базиса

$$\min_{y\in\mathbb{R}^n, z\in\mathbb{R}^n} c^\top (y - z)$$

s.t. $Ay - Az \le b$ (Фаза-2 (главная задача ЛП))

$y \ge 0, z \ge 0$

$$\min_{\xi\in\mathbb{R}^m, y\in\mathbb{R}^n, z\in\mathbb{R}^n} \sum_{i=1}^{m} \xi_i$$

s.t. $Ay - Az \le b + \xi$ (Фаза-1)

$y \ge 0, z \ge 0, \xi \ge 0$

- Если Фаза-2 (главная задача ЛП) имеет допустимое решение, то оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю).
  Доказательство: тривиальная проверка.

Поиск начального допустимого базиса

$$\min_{y \in \mathbb{R}^n, z \in \mathbb{R}^n} c^\top (y - z)$$

s.t. $Ay - Az \leq b$ (Фаза-2 (главная задача ЛП))

$y \geq 0, z \geq 0$

$$\min_{\xi \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^n} \sum_{i=1}^{m} \xi_i$$

s.t. $Ay - Az \leq b + \xi$ (Фаза-1)

$y \geq 0, z \geq 0, \xi \geq 0$

- Если Фаза-2 (главная задача ЛП) имеет допустимое решение, то оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю).
  Доказательство: тривиальная проверка.
- Если оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю), то мы получаем допустимый базис для Фаза-2.
  Доказательство: тривиальная проверка.

Поиск начального допустимого базиса

$$\min_{y \in \mathbb{R}^n, z \in \mathbb{R}^n} c^\top (y - z)$$

s.t. $Ay - Az \leq b$ (Фаза-2 (главная задача ЛП))

$y \geq 0, z \geq 0$

$$\min_{\xi \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^n} \sum_{i=1}^{m} \xi_i$$

s.t. $Ay - Az \leq b + \xi$ (Фаза-1)

$y \geq 0, z \geq 0, \xi \geq 0$

- Если Фаза-2 (главная задача ЛП) имеет допустимое решение, то оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю).
  Доказательство: тривиальная проверка.
- Если оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю), то мы получаем допустимый базис для Фаза-2.
  Доказательство: тривиальная проверка.

Поиск начального допустимого базиса

$$\min_{y \in \mathbb{R}^n, z \in \mathbb{R}^n} c^\top (y - z)$$

s.t. $Ay - Az \leq b$ (Фаза-2 (главная задача ЛП))

$y \geq 0, z \geq 0$

$$\min_{\xi \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^n} \sum_{i=1}^{m} \xi_i$$

s.t. $Ay - Az \leq b + \xi$ (Фаза-1)

$y \geq 0, z \geq 0, \xi \geq 0$

- Если Фаза-2 (главная задача ЛП) имеет допустимое решение, то оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю).
  Доказательство: тривиальная проверка.
- Если оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю), то мы получаем допустимый базис для Фаза-2.
  Доказательство: тривиальная проверка.

- Теперь мы знаем, что если мы можем решить задачу Фаза-1, то мы либо найдём начальную точку для симплекс-метода в исходной задаче (если переменные ξ_i равны нулю), либо проверим, что исходная задача не имеет допустимого решения (если переменные ξ_i не равны нулю).

Поиск начального допустимого базиса

$$\min_{y \in \mathbb{R}^n, z \in \mathbb{R}^n} c^\top (y - z)$$

s.t. $Ay - Az \leq b$ (Фаза-2 (главная задача ЛП))

$y \geq 0, z \geq 0$

$$\min_{\xi \in \mathbb{R}^m, y \in \mathbb{R}^n, z \in \mathbb{R}^n} \sum_{i=1}^{m} \xi_i$$

s.t. $Ay - Az \leq b + \xi$ (Фаза-1)

$y \geq 0, z \geq 0, \xi \geq 0$

- Если Фаза-2 (главная задача ЛП) имеет допустимое решение, то оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю).
  Доказательство: тривиальная проверка.
- Если оптимум Фаза-1 равен нулю (т.е. все переменные ξ_i равны нулю), то мы получаем допустимый базис для Фаза-2.
  Доказательство: тривиальная проверка.

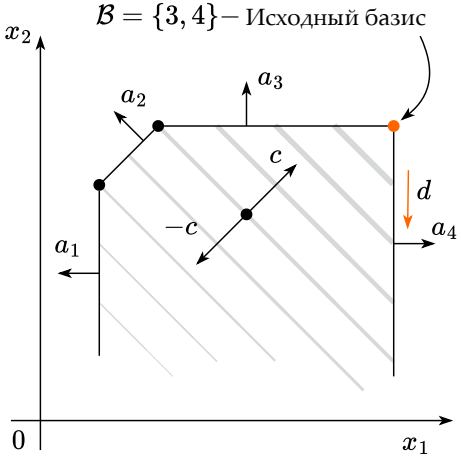
- Теперь мы знаем, что если мы можем решить задачу Фаза-1, то мы либо найдём начальную точку для симплекс-метода в исходной задаче (если переменные ξ_i равны нулю), либо проверим, что исходная задача не имеет допустимого решения (если переменные ξ_i не равны нулю).
- Но как решить задачу Фаза-1? Она имеет допустимое базисное решение (задача имеет $2n + m$ переменных, и точка ниже гарантирует, что $2n + m$ неравенств удовлетворяются как равенства (активны).) . . .

$$z = 0 \quad y = 0 \quad \xi_i = \max(0, -b_i)$$

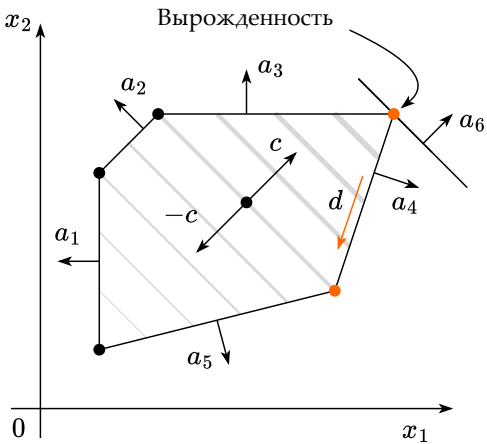
Сходимость симплекс-метода

Неограниченное бюджетное множество

В этом случае не найдётся ни одного положительного μ_j.

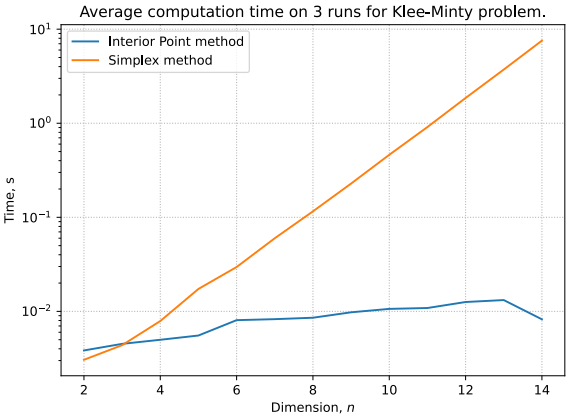


Вырожденность вершин



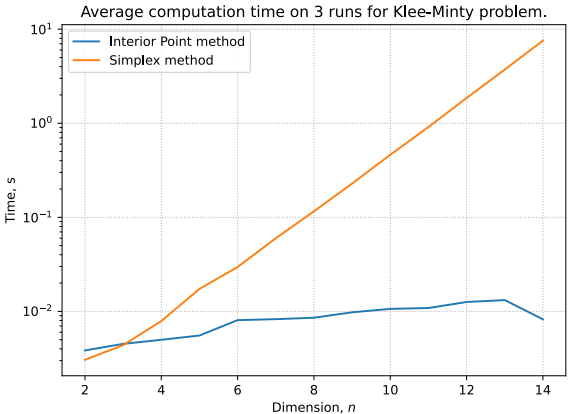
Случаи вырожденности требуют особого рассмотрения. В отсутствие вырожденности на каждой итерации гарантируется монотонное убывание значения целевой функции.

Экспоненциальная сложность симплекс-метода



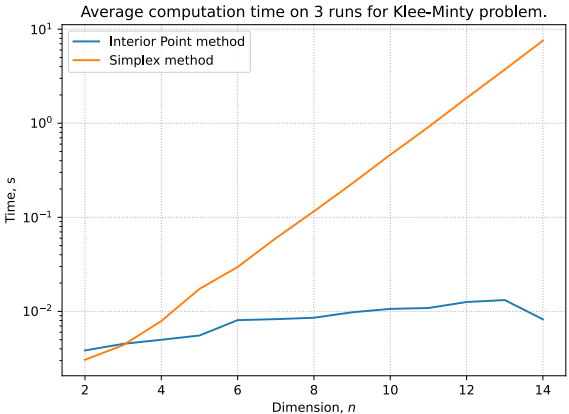
- Много прикладных задач может быть сформулировано в виде задач линейного программирования.

Экспоненциальная сложность симплекс-метода



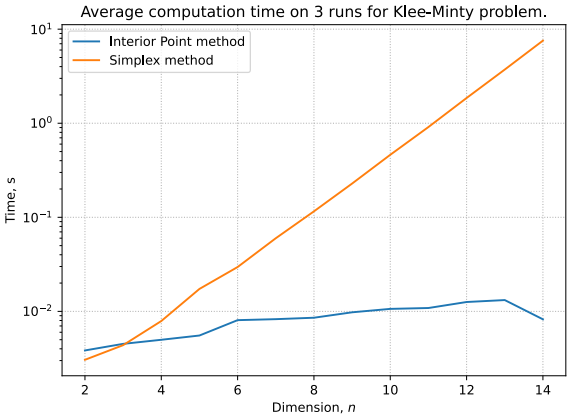
- Много прикладных задач может быть сформулировано в виде задач линейного программирования.
- Симплекс-метод прост в своей основе, но в худшем случае может работать экспоненциально долго.

Экспоненциальная сложность симплекс-метода



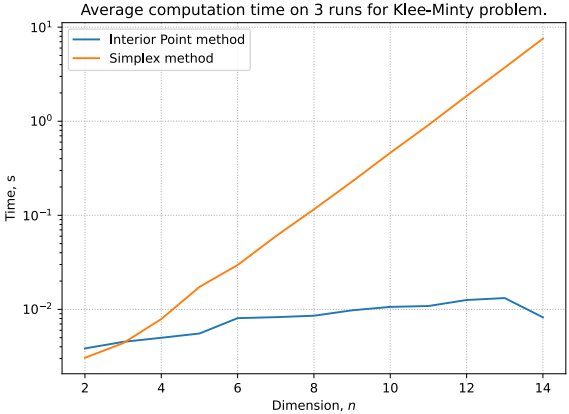
- Много прикладных задач может быть сформулировано в виде задач линейного программирования.
- Симплекс-метод прост в своей основе, но в худшем случае может работать экспоненциально долго.
- Метод эллипсоидов Хачияна (1979) стал первым алгоритмом с доказанной полиномиальной сложностью для задач ЛП. Однако он обычно работает медленнее, чем симплекс-метод в реальных небольших задачах.

Экспоненциальная сложность симплекс-метода



- Много прикладных задач может быть сформулировано в виде задач линейного программирования.
- Симплекс-метод прост в своей основе, но в худшем случае может работать экспоненциально долго.
- Метод эллипсоидов Хачияна (1979) стал первым алгоритмом с доказанной полиномиальной сложностью для задач ЛП. Однако он обычно работает медленнее, чем симплекс-метод в реальных небольших задачах.
- Основной прорыв — метод Кармаркара (1984) для решения задач ЛП с использованием метода внутренней точки.

Экспоненциальная сложность симплекс-метода



- Много прикладных задач может быть сформулировано в виде задач линейного программирования.
- Симплекс-метод прост в своей основе, но в худшем случае может работать экспоненциально долго.
- Метод эллипсоидов Хачияна (1979) стал первым алгоритмом с доказанной полиномиальной сложностью для задач ЛП. Однако он обычно работает медленнее, чем симплекс-метод в реальных небольших задачах.
- Основной прорыв — метод Кармаркара (1984) для решения задач ЛП с использованием метода внутренней точки.
- Методы внутренней точки являются последним словом в этой области. Тем не менее, для типовых задач ЛП качественные реализации симплекс-метода и методов внутренней точки показывают схожую производительность.

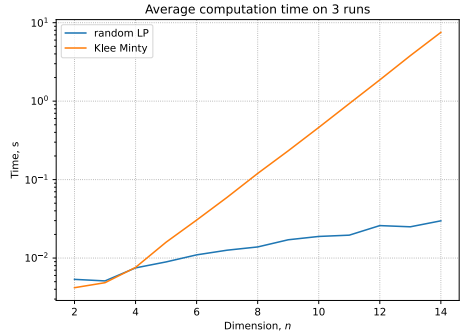
Пример Klee Minty

Так как число вершин конечно, сходимость алгоритма гарантирована (за исключением вырожденных случаев, которые здесь не рассматриваются). Тем не менее, сходимость может быть экспоненциально медленной из-за потенциально большого числа вершин. Существует пример, в котором симплекс-метод вынужден пройти через все вершины многогранника.

В следующей задаче симплекс-метод должен проверить $2^n - 1$ вершин с $x_0 = 0$.

$$\max_{x \in \mathbb{R}^n} 2^{n-1}x_1 + 2^{n-2}x_2 + \cdots + 2x_{n-1} + x_n$$

$$\begin{aligned}
\text{s.t. } & x_1 \leq 5 \\
& 4x_1 + x_2 \leq 25 \\
& 8x_1 + 4x_2 + x_3 \leq 125 \\
& \ldots \\
& 2^n x_1 + 2^{n-1}x_2 + 2^{n-2}x_3 + \ldots + x_n \leq 5^n \\
& x \geq 0
\end{aligned}$$



Двойственность в линейном программировании

Двойственная задача к ЛП

Прямая задача:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} c^\top x \\ \text{s.t. } &Ax = b \\ &x_i \geq 0,\ i = 1, \dots, n \end{aligned} \tag{3}$$

Двойственная задача к ЛП

Прямая задача:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} c^\top x \\ \text{s.t. } &Ax = b \\ &x_i \geq 0,\ i = 1, \dots, n \end{aligned} \tag{3}$$

Условия KKT для оптимума x^*, ν^*, λ^*:

$$L(x, \nu, \lambda) = c^T x + \nu^T (Ax - b) - \lambda^T x$$
$$-A^T \nu^* + \lambda^* = c$$
$$Ax^* = b$$
$$x^* \succeq 0$$
$$\lambda^* \succeq 0$$
$$\lambda_i^* x_i^* = 0$$

Двойственная задача к ЛП

Прямая задача:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} c^\top x \\ \text{s.t. } &Ax = b \\ &x_i \geq 0,\ i = 1, \dots, n \end{aligned} \tag{3}$$

Условия KKT для оптимума x^*, ν^*, λ^*:

$$\begin{aligned} &L(x, \nu, \lambda) = c^T x + \nu^T (Ax - b) - \lambda^T x \\ &-A^T \nu^* + \lambda^* = c \\ &Ax^* = b \\ &x^* \succeq 0 \\ &\lambda^* \succeq 0 \\ &\lambda_i^* x_i^* = 0 \end{aligned}$$

Двойственная задача имеет вид:

$$\begin{aligned} &\max_{\nu \in \mathbb{R}^m} -b^\top \nu \\ \text{s.t. } &-A^T \nu \preceq c \end{aligned} \tag{4}$$

Найдите двойственную задачу к задаче выше (она должна совпасть с исходной задачей ЛП). Также запишите условия KKT для двойственной задачи и убедитесь, что они идентичны условиям KKT прямой задачи.

Сильная двойственность в линейном программировании

Theorem

(i) Если одна из задач (3) или (4) имеет (конечное) решение, то и другая тоже, причём значения целевых функций совпадают.

Сильная двойственность в линейном программировании

Theorem

(i) Если одна из задач (3) или (4) имеет (конечное) решение, то и другая тоже, причём значения целевых функций совпадают.

(ii) Если одна из задач (3) или (4) неограничена, то другая задача недопустима.

Сильная двойственность в линейном программировании

Theorem

(i) Если одна из задач (3) или (4) имеет (конечное) решение, то и другая тоже, причём значения целевых функций совпадают.

(ii) Если одна из задач (3) или (4) неограничена, то другая задача недопустима.

Сильная двойственность в линейном программировании

Theorem

(i) Если одна из задач (3) или (4) имеет (конечное) решение, то и другая тоже, причём значения целевых функций совпадают.
(ii) Если одна из задач (3) или (4) неограничена, то другая задача недопустима.

ДОКАЗАТЕЛЬСТВО. Для (i) предположим, что (3) имеет конечное оптимальное решение x^*. Из условий KKT следует, что существуют оптимальные векторы λ^* и ν^* такие, что (x^*, ν^*, λ^*) удовлетворяет KKT. Мы отметили выше, что условия KKT для (3) и (4) эквивалентны. Более того,
$c^T x^* = (-A^T \nu^* + \lambda^*)^T x^* = -(\nu^*)^T A x^* = -b^T \nu^*$, что и требовалось доказать.

Симметричное рассуждение выполняется, если мы начнём с предположения, что двойственная задача (4) имеет решение.

Сильная двойственность в линейном программировании

Theorem

(i) Если одна из задач (3) или (4) имеет (конечное) решение, то и другая тоже, причём значения целевых функций совпадают.
(ii) Если одна из задач (3) или (4) неограничена, то другая задача недопустима.

ДОКАЗАТЕЛЬСТВО. Для (i) предположим, что (3) имеет конечное оптимальное решение x^*. Из условий KKT следует, что существуют оптимальные векторы λ^* и ν^* такие, что (x^*, ν^*, λ^*) удовлетворяет KKT. Мы отметили выше, что условия KKT для (3) и (4) эквивалентны. Более того, $c^T x^* = (-A^T \nu^* + \lambda^*)^T x^* = -(\nu^*)^T A x^* = -b^T \nu^*$, что и требовалось доказать.

Симметричное рассуждение выполняется, если мы начнём с предположения, что двойственная задача (4) имеет решение.

Для доказательства (ii) предположим, что прямая задача неограничена, то есть существует последовательность точек x_k, $k = 1, 2, 3, \ldots$ таких, что

$$c^T x_k \downarrow -\infty, \quad Ax_k = b, \quad x_k \geq 0.$$

Сильная двойственность в линейном программировании

Theorem

(i) Если одна из задач (3) или (4) имеет (конечное) решение, то и другая тоже, причём значения целевых функций совпадают.
(ii) Если одна из задач (3) или (4) неограничена, то другая задача недопустима.

ДОКАЗАТЕЛЬСТВО. Для (i) предположим, что (3) имеет конечное оптимальное решение x^*. Из условий KKT следует, что существуют оптимальные векторы λ^* и ν^* такие, что (x^*, ν^*, λ^*) удовлетворяет KKT. Мы отметили выше, что условия KKT для (3) и (4) эквивалентны. Более того, $c^T x^* = (-A^T \nu^* + \lambda^*)^T x^* = -(\nu^*)^T A x^* = -b^T \nu^*$, что и требовалось доказать.

Симметричное рассуждение выполняется, если мы начнём с предположения, что двойственная задача (4) имеет решение.

Для доказательства (ii) предположим, что прямая задача неограничена, то есть существует последовательность точек x_k, $k = 1, 2, 3, \ldots$ таких, что

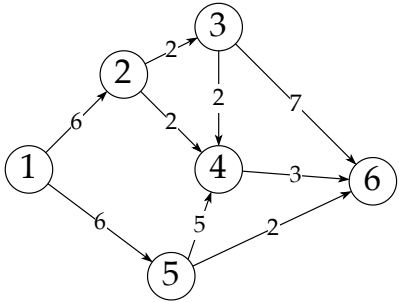
$$c^T x_k \downarrow -\infty, \quad Ax_k = b, \quad x_k \geq 0.$$

Предположим также, что двойственная задача Equation 4 допустима, т.е. существует вектор $\bar{\nu}$ такой, что $-A^T \bar{\nu} \leq c$. Из последнего неравенства совместно с $x_k \geq 0$ получаем $-\bar{\nu}^T A x_k \leq c^T x_k$, и следовательно

$$-\bar{\nu}^T b = -\bar{\nu}^T A x_k \leq c^T x_k \downarrow -\infty,$$

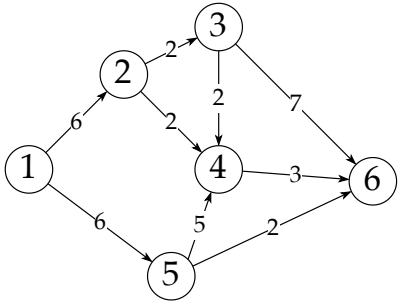
Максимальный поток — минимальный разрез

Пример задачи о максимальном потоке



Узлы представляют маршрутизаторы, рёбра — каналы связи; каждому ребру сопоставлена пропускная способность — например, узел 1 может передавать узлу 2 данные со скоростью до 6 Мбит/с, узел 2 узлу 4 — до 2 Мбит/с и т.д.

Пример задачи о максимальном потоке

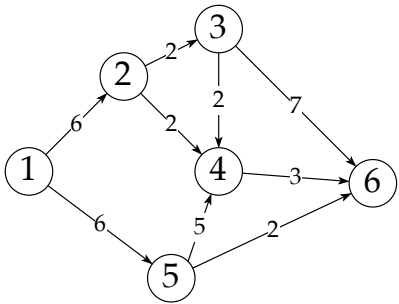


Узлы представляют маршрутизаторы, рёбра — каналы связи; каждому ребру сопоставлена пропускная способность — например, узел 1 может передавать узлу 2 данные со скоростью до 6 Мбит/с, узел 2 узлу 4 — до 2 Мбит/с и т.д.

Вопрос:

- Сеть из узлов и рёбер представляет каналы связи, каждый с заданной пропускной способностью.

Пример задачи о максимальном потоке

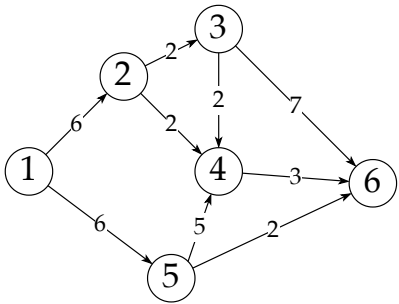


Узлы представляют маршрутизаторы, рёбра — каналы связи; каждому ребру сопоставлена пропускная способность — например, узел 1 может передавать узлу 2 данные со скоростью до 6 Мбит/с, узел 2 узлу 4 — до 2 Мбит/с и т.д.

Вопрос:

- Сеть из узлов и рёбер представляет каналы связи, каждый с заданной пропускной способностью.
- Пример: может ли узел 1 (источник) передавать данные узлу 6 (сток) со скоростью 6 Мбит/с? 12 Мбит/с? Какова максимальная скорость?

Пример задачи о максимальном потоке

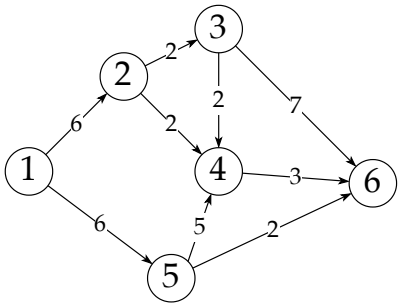


Узлы представляют маршрутизаторы, рёбра — каналы связи; каждому ребру сопоставлена пропускная способность — например, узел 1 может передавать узлу 2 данные со скоростью до 6 Мбит/с, узел 2 узлу 4 — до 2 Мбит/с и т.д.

Вопрос:

- Сеть из узлов и рёбер представляет каналы связи, каждый с заданной пропускной способностью.
- Пример: может ли узел 1 (источник) передавать данные узлу 6 (сток) со скоростью 6 Мбит/с? 12 Мбит/с? Какова максимальная скорость?

Пример задачи о максимальном потоке



Узлы представляют маршрутизаторы, рёбра — каналы связи; каждому ребру сопоставлена пропускная способность — например, узел 1 может передавать узлу 2 данные со скоростью до 6 Мбит/с, узел 2 узлу 4 — до 2 Мбит/с и т.д.

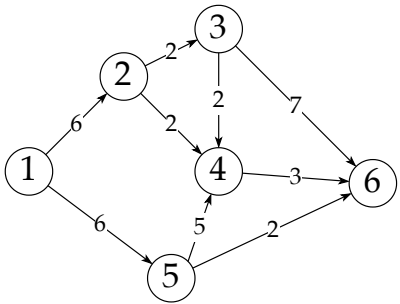
Вопрос:

- Сеть из узлов и рёбер представляет каналы связи, каждый с заданной пропускной способностью.
- Пример: может ли узел 1 (источник) передавать данные узлу 6 (сток) со скоростью 6 Мбит/с? 12 Мбит/с? Какова максимальная скорость?

Матрица пропускных способностей:

$$C = \begin{bmatrix} 0 & 6 & 0 & 0 & 6 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 5 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Пример задачи о максимальном потоке



Узлы представляют маршрутизаторы, рёбра — каналы связи; каждому ребру сопоставлена пропускная способность — например, узел 1 может передавать узлу 2 данные со скоростью до 6 Мбит/с, узел 2 узлу 4 — до 2 Мбит/с и т.д.

Вопрос:

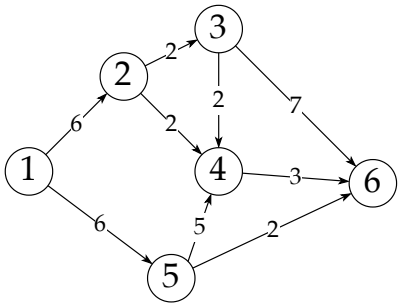
- Сеть из узлов и рёбер представляет каналы связи, каждый с заданной пропускной способностью.
- Пример: может ли узел 1 (источник) передавать данные узлу 6 (сток) со скоростью 6 Мбит/с? 12 Мбит/с? Какова максимальная скорость?

Матрица пропускных способностей:

$$C = \begin{bmatrix} 0 & 6 & 0 & 0 & 6 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 5 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Матрица потока: $X[i, j]$ обозначает поток из узла i в узел j.

Пример задачи о максимальном потоке



Узлы представляют маршрутизаторы, рёбра — каналы связи; каждому ребру сопоставлена пропускная способность — например, узел 1 может передавать узлу 2 данные со скоростью до 6 Мбит/с, узел 2 узлу 4 — до 2 Мбит/с и т.д.

Вопрос:

- Сеть из узлов и рёбер представляет каналы связи, каждый с заданной пропускной способностью.
- Пример: может ли узел 1 (источник) передавать данные узлу 6 (сток) со скоростью 6 Мбит/с? 12 Мбит/с? Какова максимальная скорость?

Матрица пропускных способностей:

$$C = \begin{bmatrix} 0 & 6 & 0 & 0 & 6 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 0 & 5 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

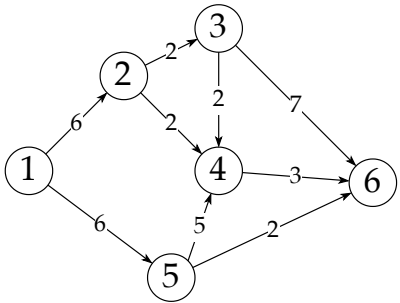
Матрица потока: $X[i, j]$ обозначает поток из узла i в узел j.

Ограничения:

$$0 \preceq X \qquad X \preceq C$$

Сохранение потока: $\sum_{j=2}^{N} X(i, j) = \sum_{k=1}^{N-1} X(k, i),\ i = 2, \ldots, N-1$

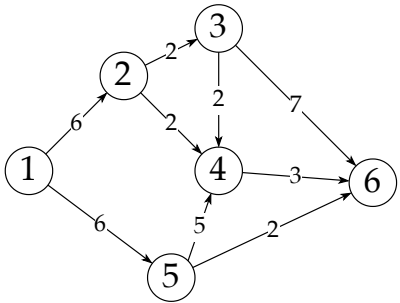
Пример задачи о максимальном потоке



В данной постановке всё, что произведено источником, поступает в сток. Поток сети — это просто сумма всего, что выходит из источника:

$$\sum_{i=2}^{N} X(1,i) \qquad \text{(Поток)}$$

Пример задачи о максимальном потоке



В данной постановке всё, что произведено источником, поступает в сток. Поток сети — это просто сумма всего, что выходит из источника:

$$\sum_{i=2}^{N} X(1,i) \qquad \text{(Поток)}$$

$$\begin{aligned} &\text{maximize } \langle X, S\rangle \\ &\text{s.t. } -X \preceq 0 \\ &\qquad X \preceq C \\ &\langle X, L_n\rangle = 0,\ n = 2, \dots, N-1, \end{aligned} \qquad \text{(Задача максимального потока)}$$

L_n состоит из единичного столбца (n) из единиц (кроме последней строки) минус единичная строка (также n) из единиц (кроме первого столбца).

$$S = \begin{bmatrix} 0 & 1 & \cdots & 1 \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}, \quad L_2 = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & -1 & \cdots & -1 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}.$$

Двойственная задача к задаче о максимальном потоке

Двойственная задача к задаче о максимальном потоке

$$\begin{aligned} &\text{minimize } \langle \Lambda, C \rangle \\ &\Lambda, \nu \\ &\text{s.t. } \Lambda + Q \succeq S \\ &\Lambda \succeq 0 \end{aligned}$$

(Двойственная задача максимального потока)

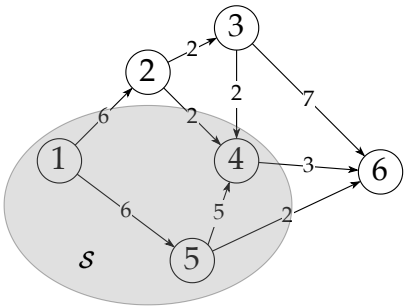
где

$$Q = \begin{bmatrix} 0 & \nu_2 & \nu_3 & \cdots & \nu_{N-1} & 0 \\ 0 & 0 & \nu_3 - \nu_2 & \cdots & \nu_{N-1} - \nu_2 & -\nu_2 \\ 0 & \nu_2 - \nu_3 & 0 & \cdots & \nu_{N-1} - \nu_3 & -\nu_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \nu_2 - \nu_{N-1} & \nu_3 - \nu_{N-1} & \cdots & 0 & -\nu_{N-1} \\ 0 & 0 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

Пример задачи о минимальном разрезе

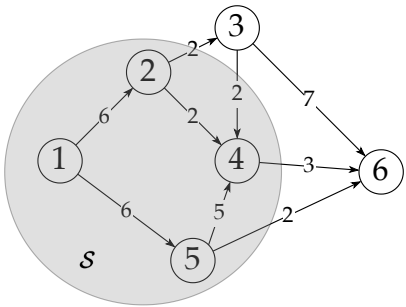
Разрез сети разделяет множество вершин на два подмножества: одно содержит источник (назовём его $\mathcal{S}$), а другое — сток. Пропускная способность разреза — это суммарная пропускная способность рёбер, выходящих из $\mathcal{S}$, то есть мы разделяем множества, «разрезая поток» по этим рёбрам.

$\mathcal{S} = \{1, 4, 5\}$



Рёбра в разрезе: $1 \to 2, 4 \to 6$ и $5 \to 6$. Пропускная способность этого разреза равна $6 + 3 + 2 = 11$.

$\mathcal{S} = \{1, 2, 4, 5\}$



Рёбра в разрезе: $2 \to 3, 4 \to 6$ и $5 \to 6$. Пропускная способность этого разреза равна $2 + 3 + 2 = 7$.

Минимальный разрез есть двойственная задача к максимальному потоку

Какова минимальная пропускная способность наименьшего разреза? Мы покажем, что она совпадает с оптимальным значением двойственной программы d^* (Двойственная задача максимального потока).

Минимальный разрез есть двойственная задача к максимальному потоку

Какова минимальная пропускная способность наименьшего разреза? Мы покажем, что она совпадает с оптимальным значением двойственной программы d^* (Двойственная задача максимального потока).

Во-первых, предположим, что $\mathcal{S}$ — допустимый разрез. Из $\mathcal{S}$ мы можем легко построить допустимую точку двойственной задачи, совпадающую по значению с пропускной способностью разреза: для $n = 1, \dots, N$ положим

$$\nu_n = \begin{cases} 1, & n \in \mathcal{S}, \\ 0, & n \notin \mathcal{S}, \end{cases} \qquad \text{и} \qquad \lambda_{i,j} = \begin{cases} \max(\nu_i - \nu_j, 0), & i \neq 1,\ j \neq N, \\ 1 - \nu_j, & i = 1, \\ \nu_i, & j = N. \end{cases}$$

Минимальный разрез есть двойственная задача к максимальному потоку

Какова минимальная пропускная способность наименьшего разреза? Мы покажем, что она совпадает с оптимальным значением двойственной программы d^* (Двойственная задача максимального потока).

Во-первых, предположим, что $\mathcal{S}$ — допустимый разрез. Из $\mathcal{S}$ мы можем легко построить допустимую точку двойственной задачи, совпадающую по значению с пропускной способностью разреза: для $n = 1, \dots, N$ положим

$$\nu_n = \begin{cases} 1, & n \in \mathcal{S}, \\ 0, & n \notin \mathcal{S}, \end{cases} \qquad \text{и} \qquad \lambda_{i,j} = \begin{cases} \max(\nu_i - \nu_j, 0), & i \neq 1,\ j \neq N, \\ 1 - \nu_j, & i = 1, \\ \nu_i, & j = N. \end{cases}$$

Заметим, что эти значения удовлетворяют ограничениям двойственной задачи, а $\lambda_{i,j}$ равно 1, если ребро $i \to j$ разрезано, и 0 в противном случае, поэтому

$$\text{capacity}(S) = \sum_{i,j} \lambda_{i,j} C_{i,j}.$$

Минимальный разрез есть двойственная задача к максимальному потоку

Какова минимальная пропускная способность наименьшего разреза? Мы покажем, что она совпадает с оптимальным значением двойственной программы d^* (Двойственная задача максимального потока).

Во-первых, предположим, что $\mathcal{S}$ — допустимый разрез. Из $\mathcal{S}$ мы можем легко построить допустимую точку двойственной задачи, совпадающую по значению с пропускной способностью разреза: для $n = 1, \dots, N$ положим

$$\nu_n = \begin{cases} 1, & n \in \mathcal{S}, \\ 0, & n \notin \mathcal{S}, \end{cases} \qquad \text{и} \qquad \lambda_{i,j} = \begin{cases} \max(\nu_i - \nu_j, 0), & i \neq 1,\ j \neq N, \\ 1 - \nu_j, & i = 1, \\ \nu_i, & j = N. \end{cases}$$

Заметим, что эти значения удовлетворяют ограничениям двойственной задачи, а $\lambda_{i,j}$ равно 1, если ребро $i \to j$ разрезано, и 0 в противном случае, поэтому

$$\text{capacity}(S) = \sum_{i,j} \lambda_{i,j} C_{i,j}.$$

Каждый разрез является допустимым, следовательно

$$d^* \leq \text{MINCUT}.$$

Минимальный разрез есть двойственная задача к максимальному потоку

Теперь покажем, что для каждого решения ν^*, λ^* двойственной задачи существует разрез с пропускной способностью не более d^*. Мы построим разрез случайным образом и покажем, что математическое ожидание пропускной способности разреза не превосходит d^* — это означает, что существует хотя бы один разрез с пропускной способностью не более d^*.

Минимальный разрез есть двойственная задача к максимальному потоку

Теперь покажем, что для каждого решения ν^*, λ^* двойственной задачи существует разрез с пропускной способностью не более d^*. Мы построим разрез случайным образом и покажем, что математическое ожидание пропускной способности разреза не превосходит d^* — это означает, что существует хотя бы один разрез с пропускной способностью не более d^*.

Пусть Z — равномерная случайная величина на $[0, 1]$. Вместе с $\lambda^*, \nu_2^*, \dots, \nu_{N-1}^*$, полученными из решения (Двойственная задача максимального потока), положим $\nu_1 = 1$ и $\nu_N = 0$. Построим разрез $\mathcal{S}$ по правилу:

$$\text{если } \nu_n^* > Z, \text{ то берём } n \in \mathcal{S}.$$

. . . Вероятность того, что конкретное ребро $i \to j$ попадёт в разрез, равна

$$\begin{aligned} P(i \in \mathcal{S}, j \notin \mathcal{S}) &= P\left(\nu_j^\star \le Z \le \nu_i^\star\right) \\ &\le \begin{cases} \max(\nu_i^\star - \nu_j^\star, 0), & 2 \le i, j \le N-1, \\ 1 - \nu_j^\star, & i = 1;\ j = 2, \dots, N-1, \\ \nu_i^\star, & i = 2, \dots, N-1;\ j = N, \\ 1, & i = 1;\ j = N. \end{cases} \\ &\le \lambda_{i,j}^\star, \end{aligned}$$

Минимальный разрез есть двойственная задача к максимальному потоку

Последнее неравенство следует непосредственно из ограничений двойственной программы (Двойственная задача максимального потока). Данный разрез случаен, поэтому его пропускная способность является случайной величиной, и её математическое ожидание равно

$$\begin{aligned}\mathbb{E}[\text{capacity}(\mathcal{S})] &= \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S}) \\ &\leq \sum_{i,j} C_{i,j} \lambda^\star_{i,j} = d^\star.\end{aligned}$$

Минимальный разрез есть двойственная задача к максимальному потоку

Последнее неравенство следует непосредственно из ограничений двойственной программы (Двойственная задача максимального потока). Данный разрез случаен, поэтому его пропускная способность является случайной величиной, и её математическое ожидание равно

$$\begin{aligned}\mathbb{E}[\text{capacity}(\mathcal{S})] &= \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S}) \\ &\leq \sum_{i,j} C_{i,j} \lambda^\star_{i,j} = d^\star.\end{aligned}$$

Следовательно, существует разрез, пропускная способность которого не превосходит $d^\star$. Это доказывает, что

$$\text{MINCUT} \leq d^\star.$$

Минимальный разрез есть двойственная задача к максимальному потоку

Последнее неравенство следует непосредственно из ограничений двойственной программы (Двойственная задача максимального потока). Данный разрез случаен, поэтому его пропускная способность является случайной величиной, и её математическое ожидание равно

$$\begin{aligned}\mathbb{E}[\text{capacity}(\mathcal{S})] &= \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S}) \\ &\leq \sum_{i,j} C_{i,j} \lambda^\star_{i,j} = d^\star.\end{aligned}$$

Следовательно, существует разрез, пропускная способность которого не превосходит $d^\star$. Это доказывает, что

$$\text{MINCUT} \leq d^\star.$$

Объединяя оба факта, получаем

$$d^\star = \text{MINCUT} = \text{MAXFLOW} = p^\star,$$

где $p^\star$ — решение прямой задачи, а равенство следует из сильной двойственности для линейного программирования.

Минимальный разрез есть двойственная задача к максимальному потоку

Последнее неравенство следует непосредственно из ограничений двойственной программы (Двойственная задача максимального потока). Данный разрез случаен, поэтому его пропускная способность является случайной величиной, и её математическое ожидание равно

$$\mathbb{E}[\text{capacity}(\mathcal{S})] = \sum_{i,j} C_{i,j} P(i \in \mathcal{S}, j \notin \mathcal{S})$$
$$\leq \sum_{i,j} C_{i,j} \lambda^\star_{i,j} = d^\star.$$

Следовательно, существует разрез, пропускная способность которого не превосходит $d^\star$. Это доказывает, что

$$\text{MINCUT} \leq d^\star.$$

Объединяя оба факта, получаем

$$d^\star = \text{MINCUT} = \text{MAXFLOW} = p^\star,$$

где $p^\star$ — решение прямой задачи, а равенство следует из сильной двойственности для линейного программирования.

Теорема о максимальном потоке и минимальном разрезе

Максимальное значение потока из s в t равно минимальной пропускной способности среди всех s-t разрезов.

Смешанное целочисленное программирование (MIP)

Сложность MIP

Рассмотрим следующую задачу смешанного целочисленного программирования (MIP):

$$
\begin{aligned}
z = 8x_1 + 11x_2 + 6x_3 + 4x_4 &\to \max_{x_1, x_2, x_3, x_4} \\
\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 &\leq 14 \\
x_i &\in \{0, 1\} \quad \forall i
\end{aligned}
\tag{5}
$$

Сложность MIP

Рассмотрим следующую задачу смешанного целочисленного программирования (MIP):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{5}$$
$$x_i \in \{0,1\} \quad \forall i$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{6}$$
$$x_i \in [0,1] \quad \forall i$$

Сложность MIP

Рассмотрим следующую задачу смешанного целочисленного программирования (MIP):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{5}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ и } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{6}$$
$$x_i \in [0,1] \quad \forall i$$

Сложность MIP

Рассмотрим следующую задачу смешанного целочисленного программирования (MIP):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{5}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ и } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{6}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ и } z = 22.$$

Сложность MIP

Рассмотрим следующую задачу смешанного целочисленного программирования (MIP):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{5}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ и } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{6}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ и } z = 22.$$

- Округление $x_3 = 0$: даёт $z = 19$.

Сложность MIP

Рассмотрим следующую задачу смешанного целочисленного программирования (MIP):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \le 14 \tag{5}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ и } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \le 14 \tag{6}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ и } z = 22.$$

- Округление $x_3 = 0$: даёт $z = 19$.
- Округление $x_3 = 1$: недопустимо.

Сложность MIP

Рассмотрим следующую задачу смешанного целочисленного программирования (MIP):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \le 14 \tag{5}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ и } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \le 14 \tag{6}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ и } z = 22.$$

- Округление $x_3 = 0$: даёт $z = 19$.
- Округление $x_3 = 1$: недопустимо.

Сложность MIP

Рассмотрим следующую задачу смешанного целочисленного программирования (MIP):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{5}$$
$$x_i \in \{0, 1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ и } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{6}$$
$$x_i \in [0, 1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ и } z = 22.$$

- Округление $x_3 = 0$: даёт $z = 19$.
- Округление $x_3 = 1$: недопустимо.

! MIP намного сложнее, чем ЛП

- Наивное округление решения, полученного для ЛП-релаксации исходной задачи MIP, может привести к недопустимому или неоптимальному решению.

Сложность MIP

Рассмотрим следующую задачу смешанного целочисленного программирования (MIP):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{5}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ и } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{6}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ и } z = 22.$$

- Округление $x_3 = 0$: даёт $z = 19$.
- Округление $x_3 = 1$: недопустимо.

! MIP намного сложнее, чем ЛП

- Наивное округление решения, полученного для ЛП-релаксации исходной задачи MIP, может привести к недопустимому или неоптимальному решению.
- Общая задача MIP является NP-трудной задачей.

Сложность MIP

Рассмотрим следующую задачу смешанного целочисленного программирования (MIP):

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{5}$$
$$x_i \in \{0,1\} \quad \forall i$$

Оптимальное решение

$$x_1 = 0, x_2 = x_3 = x_4 = 1, \text{ и } z = 21.$$

Упростим её до:

$$z = 8x_1 + 11x_2 + 6x_3 + 4x_4 \to \max_{x_1,x_2,x_3,x_4}$$
$$\text{s.t. } 5x_1 + 7x_2 + 4x_3 + 3x_4 \leq 14 \tag{6}$$
$$x_i \in [0,1] \quad \forall i$$

Оптимальное решение

$$x_1 = x_2 = 1, x_3 = 0.5, x_4 = 0, \text{ и } z = 22.$$

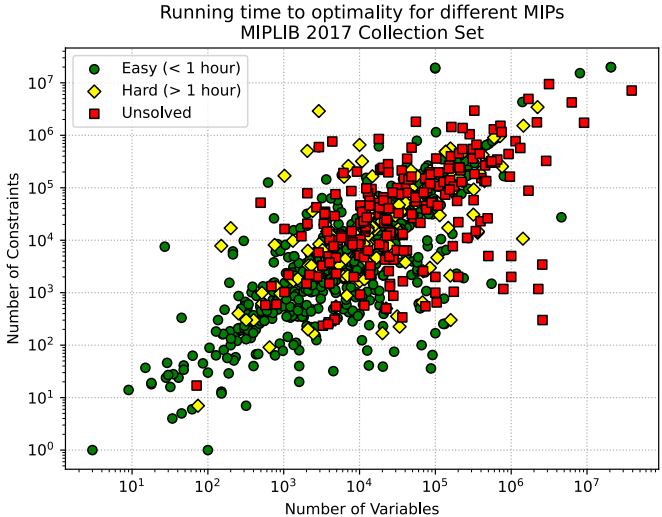
- Округление $x_3 = 0$: даёт $z = 19$.
- Округление $x_3 = 1$: недопустимо.

! MIP намного сложнее, чем ЛП

- Наивное округление решения, полученного для ЛП-релаксации исходной задачи MIP, может привести к недопустимому или неоптимальному решению.
- Общая задача MIP является NP-трудной задачей.
- Однако, если матрица коэффициентов MIP является полностью унимодулярной матрицей, то она может быть решена за полиномиальное время.

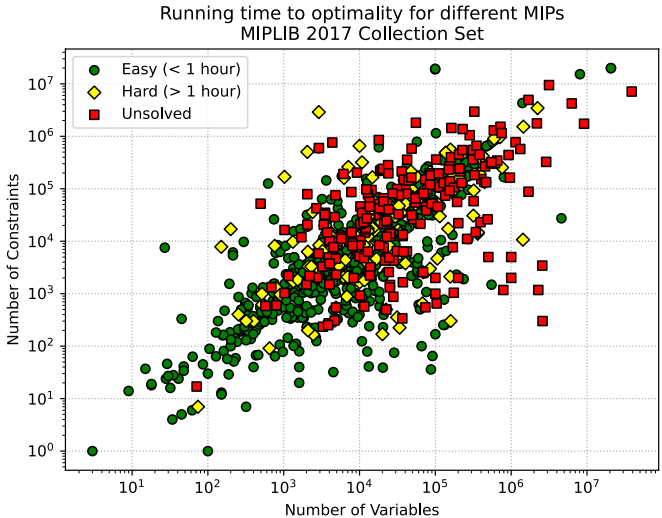
Непредсказуемая сложность MIP

- Трудно предсказать, что будет решено быстро, а что потребует много времени



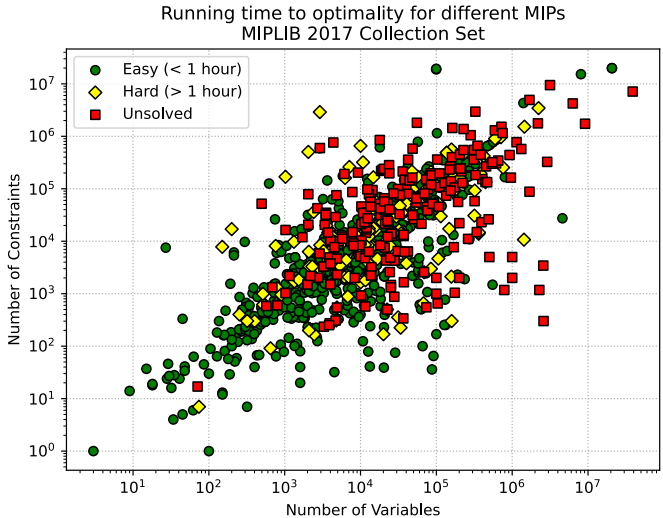
Непредсказуемая сложность MIP

- Трудно предсказать, что будет решено быстро, а что потребует много времени
- 🔗Датасет



Непредсказуемая сложность MIP

- Трудно предсказать, что будет решено быстро, а что потребует много времени
- 🔗Датасет
- Код



Прогресс аппаратного vs программного обеспечения

Что бы вы выбрали, если предположить, что вопрос поставлен корректно (вы можете скомпилировать ПО для любого оборудования, и задача в обоих случаях одна и та же)? Мы рассмотрим период с 1992 по 2023 год.

Аппаратное обеспечение

Решение MIP с использованием старого ПО на современном оборудовании

Программное обеспечение

Решение MIP с использованием современного ПО на старом оборудовании

Прогресс аппаратного vs программного обеспечения

Что бы вы выбрали, если предположить, что вопрос поставлен корректно (вы можете скомпилировать ПО для любого оборудования, и задача в обоих случаях одна и та же)? Мы рассмотрим период с 1992 по 2023 год.

Аппаратное обеспечение

Решение MIP с использованием старого ПО на современном оборудовании

$\approx 1\,664\,510 \times$ ускорение

Закон Мура утверждает, что вычислительная мощность удваивается каждые 18 месяцев.

Программное обеспечение

Решение MIP с использованием современного ПО на старом оборудовании

$\approx 2\,349\,000 \times$ ускорение

Р. Биксби провёл масштабный эксперимент по сравнению производительности всех версий CPLEX с 1992 по 2007 год и измерил общий прогресс ПО (29000 раз), позже (в 2009 году) он стал одним из основателей Gurobi Optimization, которое дало дополнительное ≈ 81 ускорение на MIP.

[2] R. Bixby report Recent study

Прогресс аппаратного vs программного обеспечения

Что бы вы выбрали, если предположить, что вопрос поставлен корректно (вы можете скомпилировать ПО для любого оборудования, и задача в обоих случаях одна и та же)? Мы рассмотрим период с 1992 по 2023 год.

Аппаратное обеспечение

Решение MIP с использованием старого ПО на современном оборудовании

$\approx 1\,664\,510 \times$ ускорение

Программное обеспечение

Решение MIP с использованием современного ПО на старом оборудовании

$\approx 2\,349\,000 \times$ ускорение

Закон Мура утверждает, что вычислительная мощность удваивается каждые 18 месяцев.

Р. Биксби провёл масштабный эксперимент по сравнению производительности всех версий CPLEX с 1992 по 2007 год и измерил общий прогресс ПО (29000 раз), позже (в 2009 году) он стал одним из основателей Gurobi Optimization, которое дало дополнительное ≈ 81 ускорение на MIP.

Оказывается, что если вам нужно решить MIP, лучше использовать старый компьютер и современные методы, чем наоборот, самый новый компьютер и методы начала 1990-х годов![2]

[2] R. Bixby report · Recent study

Источники

- Теория оптимизации (MATH4230) курс @ CUHK, профессор Тейюн Цень