1 Постановка задачи

Даны заготовки с длинной $L = 11.7$. Необходимо произвести из заготовок 11 видов изделий в количестве, заданном табл. 1, минимизируя отходы. При этом два изделия имеют одинаковую длину.

Наименование (i)	Длина (b_i)	Количество (n_i)
1	0.6	249
2	0.68	60
3	0.83	97
4	1.61	76
5	1.67	72
6	1.79	18
7	2.8	43
8	3.25	5424
9	3.25	450
10	3.7	515
11	3.95	28

Таблица 1: Параметры производства изделий

1.1 Формализация задачи

Для изготовления изделий из заготовок необходимо разрезать последние на части с соответсвующими длинами изделий. Для этих целей составляется таблица A вариантов разреза одной заготовки. В нашем случае это сделано программно перебором соответсвующих коэффициентов c_i в линейной комбинации $plan = \sum_{i=1}^{11} c_i * b_i$ при условии $plan \leq L$, где $c_i = \overline{0, \frac{L}{b_i}}$.

Пусть $k_j \in \mathbb{N}$ - необходимое количество заготовок для реализации варианта разреза j. Общее количество заготовок $k = \sum_j k_j$.

Для решения данной задачи используется симплекс-метод. Функция цели - минимум всех отходов после разрезов $\min \sum_j k_j * (L - plan)$. Ограничения задаются транспонированием таблицы A вариантов разреза и вектором свободных членов в виде $\{n_i\}_{i=\overline{1,11}}$, то есть $A^T * K = B$, где K, B - вектора количества заготовок и требуемого количества изделий соответсвенно.

1.2 Минимизация заготовок и отходов

Для решения данной задачи используется учет всевозможных вариантов разреза, даже совершенно не оптимальных, так как при подборе оптимального варианта выполнения плана на неоптимальных вариантах разреза должно стоять минимальное или нулевое количество их повторений. То есть данных подход совершенно не влияет на оптимальное решение задачи.

В рамках данной задачи могут существовать две эквивалентные постановки: минимизация количества отходов и минимизация числа необходимых заготовок. При этом для первого случая оптимальным будет такой план, при котором все ограничения выполняются в виде неравенств ввиду, то есть $A^T * K = B$ в указанных выше переменных. А для задачи минимизации числа заготовок можно использовать ограничения вида $A^T * K \geq B$.

1.3 Формализация задачи для постановки с двойным количеством заготовок для создания изделий

Если рассматривать ситуацию, что каждое изделение состоит из двух частей одинаковой длины, то к указанной постановке задачи стоит добавить, что вектор свободных членов B домножается покомпонентно на 2. Затем задача решается аналогично.

2 Исследование применимости метода

Алгоритм симплекс-метода применим к задачам линейного программирования на нахождение минимума. Метод работает на задачах в канонической форме при всяких вещественных значениях компонент $A \in \mathbb{R}_{m\times n}, b \in \mathbb{R}_m, c \in \mathbb{R}_n$. Матрица A должна иметь ранг m, что гарантирует наличие хотя бы одного опорного вектора.

Поставленная задача является задачей линейного программирования, так как ее целевая функция и ограничения имеют линейный вид. Ранг рассматриваемой матрицы был посчитан встроенным в библиотеку методом *numpy.linalg.matrix_rank* и равен 11 так же, как и количество строк данной матрицы. Алгоритмы метода приведения задачи к каноническому виду и симплекс-метода описаны в соответсвующей лабораторной работе и применимы к данной задаче.

3 Практическое решение задач

3.1 Задача с учетом двух равнодлинных изделий

При решении данной задачи было полученно минимальное количество заготовок, необходимых для выполнения плана, равное $k = 2153$. Всего вариантов разреза было получено 26470.

План разреза	Используемое количество
(0, 0, 0, 0, 0, 0, 3, 1, 0, 0, 0)	15
(0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0)	1486
(3, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0)	25
(2, 1, 0, 0, 0, 0, 0, 3, 0, 0, 0)	60
(0, 0, 2, 0, 0, 0, 0, 3, 0, 0, 0)	49
(0, 0, 0, 1, 0, 0, 0, 3, 0, 0, 0)	77
(0, 0, 0, 0, 1, 0, 0, 3, 0, 0, 0)	73
(0, 0, 0, 0, 0, 1, 0, 3, 0, 0, 0)	18
(0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0)	150
(0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0)	172
(2, 0, 0, 0, 1, 0, 0, 2, 0, 0, 1)	28

Таблица 2: Оптимальные варианты разрезов

3.2 Задача с учетом включения в изделие двух деталей равной длины

При решении данной задачи было полученно минимальное количество заготовок, необходимых для выполнения плана, равное $k = 4302$. Всего вариантов разреза было получено 26470.

План разреза	Используемое количество
(0, 0, 0, 0, 0, 0, 3, 1, 0, 0, 0)	29
(0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0)	2972
(3, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0)	49
(2, 1, 0, 0, 0, 0, 0, 3, 0, 0, 0)	120
(0, 0, 2, 0, 0, 0, 0, 3, 0, 0, 0)	98
(0, 0, 0, 1, 0, 0, 0, 3, 0, 0, 0)	153
(0, 0, 0, 0, 1, 0, 0, 3, 0, 0, 0)	145
(0, 0, 0, 0, 0, 1, 0, 3, 0, 0, 0)	36
(0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0)	300
(0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0)	344
(2, 0, 0, 0, 1, 0, 0, 2, 0, 0, 1)	56

Таблица 3: Оптимальные варианты разрезов

4 Обоснование результатов

В лабораторной работе по симплекс-методу была доказана обоснованность найденного результата. В данной лабораторной работе функция цели имеет достаточно большое число переменных, поэтому аналитически обосновать результаты не представляется возможным.

Критерий оптимальности проверяется каждый шаг, и алгоритм останавливается, как только данный критерий выполнен. Следовательно, можно гарантировать, что найденный план оптимален без каких-либо дополнительных проверок.

Также при проверке результата с группой, решавшей задачу минимизации используемого числа заготовок, была подтверждена оптимальность найденного решения. При этом число заготовок, найденных в обоих постановках задачи, различается на 1, что объясняется округлением величин при подсчете до целых чисел.

5 Выводы

При решении задачи было получено достаточно большое количество вариантов разреза, то есть число линейных ограничений для решения данной задачи может поддерживаться только программно. При таких условиях алгоритм симплекс-метода показал себя эффективным по времени и устойчивым к большим объемам данных. Однако симплекс-метод не рассчитан на нахождение целочисленного решения, что может вызывать нахождение не самого оптимального решения при округлении дробных чисел из всех возможных целочисленных решений.

6 Приложения

URL: Выполненная лабораторная работа на GitHub
https://github.com/ThinkingFrog/OptimizationMethods/tree/main/LAB6