Математические модели и методы принятия оптимальных решений

Береснев Владимир Леонидович

ФИТ НГУ

21 сентября 2025 г.

Оглавление

Глава 1

Введение

Определение

Организованные системы — системы, в которых решения принимаются «сознательно». Примеры таких систем: люди, промышленные предприятия, магазины.

Примеры задач:

- Где построить магазин, чтобы получать наибольшую прибыль?
- Сколько производить деталей на заводе, чтобы отношение между доходом и выручкой было наибольшим?

Примерно до второй мировой войны все сложные решения принимались лишь на основе опыта и здравого смысла. Однако позже появились сложные системы, в которых опыта и здравого смысла оказалось недостаточно. Тогда же появилась **идея** рассматривать числовые характеристики систем для принятия решения, при этом должны использоваться *математические модели* — упрощённые, но адекватные описания реальной жизни.

1.1 Построение математических моделей

Математические модели строятся на основании *исходных данных* — конкретных проблем в конкретных жизненных ситуациях.

Примерный алгоритм построения математических моделей

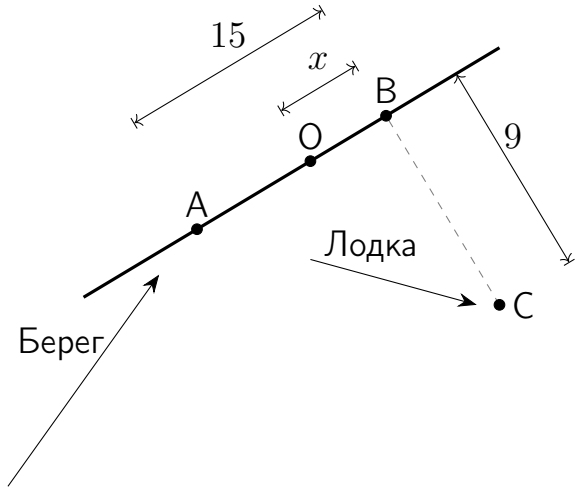
1. Нужно понять, *что будем оптимизировать?* По каким критериям будет оценивать решения? Например, если нас спрашивают, где построить магазин, то

нам нужно понять, как выбрать для этого место. Нужно, чтобы была максимальная прибыль или чтобы была наибольшая удалённость от конкурентов? Если мы закупаем детали для предприятия, то нужно узнать, хочется ли нам наибольшую прибыль или же минимальные издержки. А может нам важно количество произведённой продукции?

2. Нужно понять, *какие характеристики существенно влияют на оптимизируемую характеристику*? Например, если наша оптимизируемая характеристика — это прибыль предприятия, то нужно определить, из чего складываются выручка и затраты.
3. *Формулировка задачи* с точным указанием всех характеристик. Каждая из характеристик должна относиться либо к *переменным*, либо к *параметрам*. Первые могут меняться (обозначаются они через $x, y, z, \dots$), а вторые — это константы (обозначаются они через $a, b, c, \dots$). Например, переменные — это выручка и затраты предприятия, а параметры — это площадь помещения, количество станков, количество работников.
4. Выбор всех обозначений и математическая запись с учётом ограничений и требований для переменных.
5. Понимание, что изначальная проблема состоит в другом, не учтены такие-то параметры, а значит нужно вернуться к самому началу...

Пример

Для примера рассмотрим следующую задачу: ваш знакомый плывёт на лодке, ему нужно попасть в определённую точку на берегу, и он спрашивает вас, куда ему нужно причалить.



Характеристики рассматриваемой системы

- C — текущее местоположение лодки;
- A — точка, в которую нужно попасть;
- B — ближайшая к лодке точка берега;

- O — точка причаливания;
- $AB = 15$ км;
- $BC = 9$ км (расстояние от лодки до берега);
- $v_{\text{по суше}} = 5$ км/ч;
- $v_{\text{лодки}} = 4$ км/ч;
- $OB = x$;
- течения воды нет, то есть скорость течения равна нулю;
- цвет лодки не существенен.

Будем оптимизировать время t

$$t = \underbrace{\frac{OC}{v_{\text{по воде}}}}_{\text{движение по воде}} + \underbrace{\frac{OA}{v_{\text{по суше}}}}_{\text{движение по суше}}$$

$$= \boxed{\frac{\sqrt{x^2+81}}{4} + \frac{15-x}{5} \to \min_x}$$

Установим **ограничение**

$$0 \le x \le 15,$$

потому что иначе решение будет точно неоптимальным.

Решение

Решим задачу, найдя нули производной

$$\frac{\mathrm{d}t}{\mathrm{d}x} = \frac{2x}{8\sqrt{x^2+81}} - \frac{1}{5} = 0,$$

$$\frac{x}{4\sqrt{x^2+81}} = \frac{1}{5},$$

$$\frac{x^2}{16(x^2+81)} = \frac{1}{25},$$

$$25x^2 = 16x^2 + 16 \cdot 81,$$

$$9x^2 = 16 \cdot 81,$$

$$x^2 = 16 \cdot 9,$$

$$x_1 = -12, \qquad x_2 = 12.$$

Решение $x_1 = -12$ не подходит ввиду ограничения выше, а вот $\boxed{x = 12}$ является ответом.

1.2 Оптимизационные (экстремальные) задачи

Определение (экстремальная задача)

Экстремальная задача формулируется следующим образом

1. Есть функция $f(x)$, значение которой нужно оптимизировать;
2. Есть набор ограничений $g_1(x) \leq b_1, g_2(x) \leq b_2, \ldots$;
3. $x \in X$, X — *множество всех решений*,

при этом нужно найти

$$\min_{x \in X} f(x) \quad \text{или} \quad \max_{x \in X} f(x).$$

Определение

Допустимые решения — множество всех значений $x \in X$, которые удовлетворяют всем ограничения $\{g_i\}$.

Определение

Допустимое решение x^* называется *оптимальным решением задачи*, если

$$\forall x \in X \qquad f(x^*) \geq f(x).$$

или то же самое

$$f(x^*) = \max_{x \in X} f(x).$$

Замечание

Ограничения $\{g_i\}$ могут быть какими угодно.

Пример

Пусть наши исходные данные это

$$f(x) = c_1 x_1 + c_2 x_2 \to \max_x,$$

$$g(x) = a x_1 + b x_2 \leq d,$$

$$x_1 \geq 0, \quad x_2 \geq 0.$$

Нужно найти оптимальные x_1 и x_2.

Определение

Будем говорить, что есть *общая задача* $\mathcal{P}$, а $p \in \mathcal{P}$ — конкретная задача, в которой у всех параметров есть конкретные значения. Например, если бы в задаче выше c_1, c_2, a, b, d были бы конкретными числами. То есть *общая задача* — это множество конкретных задач.

Определение

Длина входа задачи — это количество ячеек в памяти, которое занимает задача с допущением, что каждое число занимает в памяти ровно одну ячейку. Будем обозначать это $|p|$.

1.3 Алгоритмы

Определение

Элементарные операции — арифметические операции и операции сравнения.

Определение

Трудоёмкость алгоритма A решения задачи $p \in \mathcal{P}$ — это количество элементарных операций, используемых в этом алгоритме. Будем обозначать это $T_A(p)$.

Замечание

Чем больше $|p|$, тем больше $T_A(p)$, поэтому будем оценивать трудоёмкость так

$$T_A(p) \leq f_A\big(|p|\big).$$

Определение

Если $f_A\big(|p|\big) = C \cdot |p|^k$, то такой алгоритм будем называть *«хорошим»* (*полиномиальным*).

Примерами задач, для которых существуют «хорошие» алгоритмы, являются математические задачи, в которых x — множество векторов (линейное и нелинейное программирование), и задачи комбинаторики (например, перестановки).

Определение (задача линейного программирования)

$$\sum_{j=1}^{n} c_j x_j \to \max_{(x_j)},$$

$$\sum_{j=1}^{n} a_{ij} x_j \leq b_i, \quad i = 1 \dots m,$$

$x_j \in \{0, 1, 2, \dots\}$ или то же самое $x_j \geq 0, \quad j = 1 \dots n.$

Распространённый частный случай: $x_j \in \{0, 1\}$.

1.4 Примеры задач

Для примера рассмотрим несколько оптимизационных задач.

1.4.1 Задача о машине на острове

На некотором острове есть машина; для работы машины нужны детали, которые могут изнашиваться. В скором времени нужно будет вызвать самолёт, который сможет доставить необходимые детали, однако максимальная масса груза ограничена. Сколько деталей какого типа нужно заказать?

1. Что будем оптимизировать? Ответ: машина должна работать максимальное время.

2. Что существенно влияет на оптимизируемую характеристику?
Пусть $i = 1 \dots n$ — вид детали.

Параметры (то есть фиксированные значения)

- $m = \{m_i\}_{i=1}^n$ — масса деталей;
- $t = \{t_i\}_{i=1}^n$ — срок службы деталей (часов, дней, месяцев — неважно);
- $a = \{a_i\}_{i=1}^n$ — количество работающих деталей в машине;
- M — максимальная масса посылки.

Переменные (то что может меняться)

- $x = \{x_i\}_{i=1}^n$ — сколько нужно взять деталей в посылку.

3. Математическая формулировка задачи
Пусть T — время работы машины, тогда

$$T = \min_{i=1\dots n} \big((x_i + a_i) \cdot t_i\big) \to \max_{(x_i)},$$

$$\sum_{i=1}^{n} m_i x_i \le M,$$

$$x_i \ge 0.$$

Первое выражение говорит о том, что мы максимизируем время работы машины T при различных (x_i), то есть нам нужно подобрать такие значения $x_1, x_2, \dots, x_n$, при которых время работы будет максимальным. Суть этого выражения заключается в том, что $x_i + a_i$ — это то, сколько деталей вида i будет после прилёта самолёта с посылкой. Если учесть, что каждая деталь типа i имеет срок службы t_i, то все детали данного типа проработают $(x_i + a_i) \cdot t_i$. Ясно, что машина перестанет работать, когда какой-то вид деталей в ней отработает свой срок службы, значит время работы машины — это минимум из времён работы всех деталей.

Второе выражение отражает ограничение задачи, которое состоит в том, что мы не можем заказать груз большей массы, чем установленная максимальная масса посылки. Третье выражение говорит о том, что количество заказываемых деталей не может быть отрицательным — оно и понятно.

1.4.2 Задача о салфетках

Пусть есть некоторое кафе, в которое каждый день ходят люди, при этом известно, сколько человек посещает кафе в каждый день недели. Каждому гостю на день выдают салфетку, которую вечером стирают. Салфетки можно стирать с помощью *быстрой* и *медленной* стирок. Первая — дорогая, но работает условно моментально, вторая — дешёвая, но выдача постиранных салфеток происходит лишь через день. Как сэкономить деньги на стирке так, чтобы всем посетителям всегда хватало салфеток?

1. Что будем оптимизировать? Ответ: нужно минимизировать затраты на стирку.

2. Что существенно влияет на оптимизируемую характеристику?
Пусть $i = 1 \dots 7$ — день недели.

Параметры

- c_1 — цена быстрой стирки;
- c_2 — цена медленной стирки;
- T — общее количество салфеток в кафе;
- p_i — количество гостей в i-ый день недели;

Переменные

- x_i — количество салфеток, отданных в быструю стирку в i-ый день недели;
- y_i — количество салфеток, отданных в медленную стирку в i-ый день недели.

3. Математическая формулировка задачи
Пусть C — затраты на стирку за неделю, тогда

$$C = \sum_{i=1}^{7} (c_1 x_i + c_2 y_i) \to \min_{(x_i),(y_i)},$$

$$x_i + y_i = p_i,$$

$$x_i \geq 0, \quad y_i \geq 0,$$

$$\begin{cases} T - y_7 \geq p_1, \\ T - y_1 \geq p_2, \\ T - y_2 \geq p_3, \\ \dots \\ T - y_5 \geq p_6, \\ T - y_6 \geq p_7. \end{cases}$$

Последние семь неравенств означают, что всем посетителям всегда хватает салфеток.

1.4.3 Задача раскроя

Вашему знакомому сантехнику нужно определённое количество коротких труб, однако в магазине можно купить только длинные. Сколько нужно купить длинных труб, чтобы их можно было раскроить на короткие? Пусть нужны

- 10 труб длиной 6 м,
- 15 труб длиной 5 м,
- 26 труб длиной 4 м,

а в магазине продаются лишь трубы длиной 13 м.

1. Что будем оптимизировать? Ответ: нужно минимизировать число покупаемых труб длиной 13 м.

2. Что существенно влияет на оптимизируемую характеристику? Рассмотрим все варианты, как можно раскроить длинную трубу на короткие

№	6 м	5 м	4 м
1	2	0	0
2	1	1	0
3	1	0	1
4	0	2	0
5	0	1	2
6	0	0	3

То есть раскроить длинную трубу на короткие можно 6 разными способами. Например, на две трубы длиной 6 метров (первая строка таблицы), на одну трубу длиной 6 метров и на одну трубу длиной 5 метров (вторая строка таблицы) и так далее.

Параметры

- x_i — количество длинных труб, раскроенных i-ым способом;

3. Математическая формулировка задачи

Пусть T — количество изрезанных длинных труб, тогда

$$T = \sum_{i=1}^{6} x_i \to \min_{(x_i)}$$

$$\begin{cases} 2x_1 + x_2 + x_3 \geq 10 \\ x_2 + 2x_4 + x_5 \geq 15 \\ x_3 + 2x_5 + 3x_6 \geq 26 \end{cases}$$

Последние три неравенства означают, что после раскроя длинных труб мы получили достаточное количество коротких: первое неравенство для труб длиной 6 метров, второе — 5 метров, третье — 4 метра.

Замечание

В общем случае задаче раскроя является NP-полной.

1.5 Свойства оптимизационных задач

Утверждение 1

$$\max_x f(x) = \max_x \Big(\big(-1\big) \cdot \big(-f(x)\big) \Big) = \underbrace{-\min_x \big(-f(x)\big)}_{\text{новая задача}}.$$

Утверждение 2 (оценка сверху, релаксированные задачи)

Если $X \subseteq X'$, то

$$\max_{x \in X} f(x) \leq \max_{x \in X'} f(x).$$

Определение

Пусть есть две общие задачи: $(\mathcal{P}) \max_{x \in X} f(x)$ и $(\mathcal{Q}) \max_{y \in Y} g(y)$. Будем говорить, что задача $\mathcal{P}$ *сводится к задаче* $\mathcal{Q}$, если

$$\forall p \in \mathcal{P} \quad \forall q \in \mathcal{Q}$$

1. существует полиномиальный алгоритм A_1, который переводит входные данные задачи p во входные данные задачи q;

$$p \quad \xrightarrow{A_1} \quad q$$

2. существует полиномиальный алгоритм A_2, с помощью которого можно из оптимального решения y^0 задачи q построить оптимальное решение x^* задачи p.

$$x^* \quad \xleftarrow{A_2} \quad y^0$$

Остаётся вопрос: как понять, что построенное решение x^* — оптимальное?

Замечание

Везде далее всегда будем оптимизировать именно $\max$, а не $\min$.

Утверждение 3 (сведение к другой задаче)

Пусть есть 2 задачи: $\mathcal{P}$ и $\mathcal{Q}$, при этом

1. x^0 — допустимое решение задачи $\mathcal{P}$,
2. y^* — оптимальное решение задачи $\mathcal{Q}$,
3. $f(x^0) \geq g(y^*)$,
4. $\forall x \in X \; \exists y \in Y \quad f(x) \leq g(y)$,

тогда x^0 — оптимальное решение задачи $\mathcal{P}$.

Доказательство

Пусть задача $\mathcal{P}$ имеет оптимальное решение x^*. По четвёртому пункту для x^* существует некоторый y^0 такой, что

$$f(x^*) \leq g(y^0). \qquad (*)$$

По условию y^* является оптимальным решением задачи $\mathcal{Q}$, значит верно следующее

$$g(y^*) \geq g(y^0). \qquad (**)$$

Составим цепочку неравенств

$$f(x^0) \overset{(3)}{\geq} g(y^*) \overset{(**)}{\geq} g(y^0) \overset{(*)}{\geq} f(x^*).$$

Таким образом имеем неравенство

$$f(x^0) \geq f(x^*)$$

хотя x^* — оптимальное решение задачи $\mathcal{P}$. Значит $x^0 = x^*$, то есть x^0 является оптимальным.

Пример

$(\mathcal{P})$

$$\sum_{j=1}^{n} c_j x_j \to \max_{(x_j)},$$

$$\sum_{j=1}^{n} a_j x_j \le b, \tag{1}$$

$$x_1 + x_2 = d, \tag{2}$$

$$x_j \ge 0, \quad j = 1 \dots n. \tag{3}$$

Заметим, что x_1 можно выразить через x_2 (и наоборот). Рассмотрим другую задачу

$(\mathcal{Q})$

$$c_1(d - y_2) + \sum_{j=2}^{n} c_j y_j \to \max_{(y_j)},$$

.

$$a_1(d - y_2) + \sum_{j=2}^{n} a_j y_j \le b,$$

.

$$y_j \ge 0, \quad j = 1 \dots n,$$

$$y_2 \le d.$$

Покажем, что задача $\mathcal{P}$ сводится к задаче $\mathcal{Q}$.

Доказательство

Пусть $y^* = \begin{pmatrix} y_1^* & \dots & y_n^* \end{pmatrix}$ — это оптимальное решение задачи $\mathcal{Q}$. Будем строить решение x^0 задачи $\mathcal{P}$ следующим образом

$$x_j^0 = \begin{cases} d - y_2, & j = 1 \\ y_j, & j > 1 \end{cases}$$

то есть $x^0 = \begin{pmatrix} d - y_2 & y_2 & y_3 & \dots & y_n \end{pmatrix}$.

1. Покажем, что x^0 — допустимое решение задачи $\mathcal{P}$. Для этого нужно показать, что оно удовлетворяет всем ограничениям. Заметим, что без условия $y_2 \le d$ значение x_1 может быть меньше нуля, а значит решение x^0 точно было бы не допустимым (ограничение 3).

   (a) Проверим ограничение 2

$$x_1^0 + x_2^0 = d - y_2^* + y_2^* = d.$$

(b) Проверим ограничение 1

$$\begin{aligned}\sum_{j=1}^{n} a_j x_j^0 &= a_1 x_1^0 + a_2 x_2^0 + \cdots + a_n x_n^0 \\ &= a_1(d - y_2^*) + a_2 y_2^* + \cdots + a_n y_n^* \\ &= a_1(d - y_2^*) + \sum_{j=2}^{n} a_j y_j^* \leq b.\end{aligned}$$

Значит x^0 удовлетворяет всем ограничениям, а поэтому x^0 — допустимое решение.

2. Мы показали, что x^0 — допустимое решение задачи $\mathcal{P}$. Осталось показать, что оно оптимальное. Для этого будем использовать утверждение 3.

   (a) Выполнимость условия $f(x^0) \geq g(y^*)$ следует из того, что при подстановке, использованной при проверке ограничения 1, получится равенство

   $$f(x^0) = g(y^*).$$

   (b) Выполнимость условия

   $$\forall x \in X \; \exists y \in Y \quad f(x) \leq g(y)$$

   следует из того, что по любому $x = (x_j)$ можно построить $y = (y_j)$ следующим образом

   $$y_j = \begin{cases} d - x_2, & j = 1 \\ x_j, & j > 1 \end{cases}$$

   И вновь, если всё аккуратно подставить, то получится равенство

   $$f(x) = g(y).$$

Таким образом мы доказали, что можно применить утверждение 3, значит $\mathcal{P} \mapsto \mathcal{Q}$.

Мы взяли задачу с n переменными и перешли от неё к задаче с $n-1$ переменными. Разве не круто?!

1.5.1 Решение задачи о салфетках

Решим задачу о салфетках с помощью сведения к другой задаче. Запишем математическую формулировку нашей исходной задачи $\mathcal{P}$

$$C = \sum_{i=1}^{7} (c_1 x_i + c_2 y_i) \to \min_{(x_i),(y_i)}$$

$$x_i + y_i = p_i,$$

$$x_i \geq 0, \quad y_i \geq 0,$$

$$\begin{cases} T - y_7 \geq p_1, \\ T - y_1 \geq p_2, \\ T - y_2 \geq p_3, \\ \dots \\ T - y_5 \geq p_6, \\ T - y_6 \geq p_7. \end{cases}$$

Новая задача

Сведём исходную задачу к другой задаче $\mathcal{Q}$, выразив x_i через y_i

$$x_i = p_i - y_i.$$

Запишем оптимизируемую характеристику в новой задаче $\mathcal{Q}$

$$\begin{aligned} C &= \sum_{i=1}^{7} (c_1 x_i + c_2 y_i) \to \min_{(x_i),(y_i)} \\ &= \sum_{i=1}^{7} \big(c_1(p_i - y_i) + c_2 y_i\big) \to \min_{(y_i)} \\ &= \sum_{i=1}^{7} \big(y_i(c_2 - c_1) + c_1 p_i\big) \to \min_{(y_i)} \\ &= \underbrace{(c_2 - c_1)}_{const,<0} \sum_{i=1}^{7} y_i + \underbrace{c_1 \sum_{i=1}^{7} p_i}_{const} \to \min_{(y_i)} \\ &\sim \sum_{i=1}^{7} y_i \to \max_{(y_i)} \end{aligned}$$

То есть с учётом всех констант наша новая задача $\mathcal{Q}$ свелась к

$$\sum_{i=1}^{7} y_i \to \max_{(y_i)}$$

$$\begin{cases} T - y_7 \geq p_1, \\ T - y_1 \geq p_2, \\ T - y_2 \geq p_3, \\ \dots \\ T - y_5 \geq p_6, \\ T - y_6 \geq p_7. \end{cases}$$

Стоит заметить, что в задаче $\mathcal{P}$ все $x_i \geq 0$, а значит нужно ввести ограничения на y_i

$$y_i \leq p_i.$$

Итого задача $\mathcal{Q}$ формулируется следующим образом

$$\sum_{i=1}^{7} y_i \to \max_{(y_i)}$$

$$\begin{cases} T - y_7 \geq p_1, \\ T - y_1 \geq p_2, \\ T - y_2 \geq p_3, \\ \dots \\ T - y_5 \geq p_6, \\ T - y_6 \geq p_7; \end{cases}$$

$$y_i \leq p_i.$$

Решение новой задачи

Нам нужно максимизировать сумму y_i, при этом есть на каждый y_i есть два ограничения сверху. Так, например для y_2 есть ограничения

$$\begin{cases} T - y_2 \geq p_3, \\ y_2 \leq p_2; \end{cases}$$

$$\begin{cases} y_2 \leq T - p_3, \\ y_2 \leq p_2. \end{cases}$$

Ясно, что если все y_i выбрать максимально возможными, то и их сумма будет максимально возможной. Максимально возможное значение y_i, которое удовлетворяет условие — это будет минимум из двух границ сверху. Например, для y_2 это будет

$$\min\{T - p_3, p_2\}.$$

Таким образом, мы уже можем записать оптимальное решение y^* задачи $\mathcal{Q}$

$$y_1^* = \min\{T - p_2, p_1\},$$

$$y_2^* = \min\{T - p_3, p_2\},$$

$$\dots$$

$$y_6^* = \min\{T - p_7, p_6\},$$

$$y_7^* = \min\{T - p_1, p_7\}.$$

Возвращение к исходной задаче

После того, как мы нашли оптимальное решение новой задачи $\mathcal{Q}$, нужно вернуться к исходной задаче $\mathcal{P}$. Её оптимальное решение x^* выражается следующим образом

$$x_i^* = p_i - y_i^*.$$

Таким образом, мы свели исходную задачу $\mathcal{P}$ к новой задаче $\mathcal{Q}$ с меньшим числом переменных, решили задачу $\mathcal{Q}$ и по её оптимальному решению построили оптимальное решение исходной задачи.

Почему x^* — оптимальное решение задачи $\mathcal{P}$? Для доказательства этого можно использовать утверждение 3.

1.6 Использование булевых переменных

Очень часто в реальных задачах встречаются самые разные логические условия. Например: «если верно ..., то должно быть верно ...». Данные условия можно записать на языке формул математической логики, однако в рамках данного курса будет удобнее, если они будут записаны с использованием алгебраических выражений. Таким образом мы сможем записать любые ограничения любых задач на языке алгебры. Для записи логических условий хорошо подходят булевы переменные. Это переменные, которые могут принимать лишь два значения — 0 и 1.

Утверждение 4 (простые условия)

Пусть у нас есть два логических условия A и B, и нам нужно записать на языке

алгебры логическое выражение «если верно A, то верно B». Для этого введём две булевы переменные x и y, смысл этих переменных будет следующий

$$x = \begin{cases} 0, & A \text{ — истина} \\ 1, & A \text{ — ложь} \end{cases}$$

$$y = \begin{cases} 0, & B \text{ — истина} \\ 1, & B \text{ — ложь} \end{cases}$$

Рассмотрим всевозможные комбинации событий A и B на языке булевых переменных.

1. «Если верно A, то верно B». На языке наших переменных это записывается как «если $x = 0$, то $y = 0$». Будем записывать это алгебраически следующим образом
$$x \geq y.$$
   - Если $x = 0$, то y не может быть равен 1, потому что $0 \ngeq 1$, значит y может равняться лишь 0.
   - Если $x = 1$, то y может равняться как 0, так и 1, поскольку $1 \geq 0$ и $1 \geq 1$.

   Значит наше неравенство по смыслу совпадает с исходным логическим условием.
2. Условие «если $x = 0$, то $y = 1$» записывается как
$$x \geq 1 - y.$$
   - Если $x = 0$, то $1 - y$ не может быть равен 1, а значит $1 - y = 0$, как следствие $y = 1$.
   - Если $x = 1$, то $1 - y$ может равняться как 0, так и 1, значит y может принимать любое значение из $\{0, 1\}$.

   Записать через $x \geq y - 1$ было бы некорректно, поскольку при $y = 0$ в правой части получилось бы отрицательное число, а при использовании булевых переменных нужно оперировать лишь 0 и 1.
3. «Если $x = 1$, то $y = 0$». По аналогии с предыдущим пунктом это записывается как
$$1 - x \geq y.$$
4. «Если $x = 1$, то $y = 1$». По аналогии с предыдущими пунктами это записывается как
$$1 - x \geq 1 - y,$$
   или то же самое
$$x \leq y.$$

Подытожим

1. «Если $x = 0$, то $y = 0$» $\qquad x \geq y.$
2. «Если $x = 0$, то $y = 1$» $\qquad x \geq 1 - y.$
3. «Если $x = 1$, то $y = 0$» $\qquad 1 - x \geq y.$
4. «Если $x = 1$, то $y = 1$» $\qquad 1 - x \geq 1 - y \quad \Longleftrightarrow \quad x \leq y.$

Утверждение 5 (сложные условия)

Пусть теперь у нас есть множества условий $\mathcal{A} = \{A_i\}$ и $\mathcal{B} = \{B_k\}$, которым мы будем сопоставлять булевы переменные x и y.

1. Пусть нам хочется алгебраически записать следующее условие: «если некоторые условия из множества $\mathcal{A}$ истины (а остальные ложны), то все условия из множества $\mathcal{B}$ истины». Введём булев вектор $x = (x_i)$, соответствующий условиям из $\mathcal{A}$. Также введём не пересекающиеся множества индексов $I^0, I^1 \subset I$, соответствующие верным и неверным условиям из $\mathcal{A}$ следующим образом

$$x_i = \begin{cases} 0, & i \in I^0 \Leftrightarrow A_i \text{ — истина} \\ 1, & i \in I^1 \Leftrightarrow A_i \text{ — ложь} \end{cases}$$

   Ещё введём булевы переменную y, которая обозначает следующее

$$y = \begin{cases} 0, & \text{все условия из } \mathcal{B} \text{ истины} \\ 1, & \text{не все условия из } \mathcal{B} \text{ истины} \end{cases}$$

   Фактически нам бы хотелось записать следующее: «если вектор x имеет такой вид, то $y = 0$». Записать алгебраически это можно следующим образом

$$\sum_{i \in I^1} (1 - x_i) + \sum_{i \in I^0} x_i \geq y.$$

   Действительно, если вектор x имеет нужный нам вид, то обе суммы равняются нулю, значит y ничего не остаётся кроме как быть равным нулю.
2. Пусть нам хочется алгебраически записать следующее условие: «если все события из $\mathcal{A}$ истины, то некоторые условия из множества $\mathcal{B}$ истины (а остальные ложны)». Введём булев вектор $y = (y_k)$, соответствующий условиям из $\mathcal{B}$. Также введём не пересекающиеся множества индексов $K^0, K^1 \subset K$, соответствующие верным и неверным условиям из $\mathcal{B}$ следующим образом

$$y_k = \begin{cases} 0, & k \in K^0 \Leftrightarrow B_k \text{ — истина} \\ 1, & k \in K^1 \Leftrightarrow B_k \text{ — ложь} \end{cases}$$

Ещё введём булевы переменную x, которая обозначает следующее

$$x = \begin{cases} 0, & \text{все условия из } \mathcal{A} \text{ истины} \\ 1, & \text{не все условия из } \mathcal{A} \text{ истины} \end{cases}$$

Фактически нам бы хотелось записать следующее: «если $x = 0$, то y имеет такой-то вид». Записать алгебраически это можно следующим образом

$$\left|K^0 \cup K^1\right| \cdot x \geq \sum_{k \in K^1} (1 - y_k) + \sum_{k \in K^0} y_k.$$

Зачем здесь нужен коэффициент $\left|K^0 \cup K^1\right|$?. Если $x = 0$, то наше условие соблюдается, поскольку в правой части тогда обе суммы должны равняться нулю, а значит y принимает нужным нам вид. Однако дело в том, что если $x \neq 0$, то y должен иметь возможность принимать любые значения (посылка ложна), однако алгебраически это не так. Теоретически правая часть может быть сколь угодно большой, а правая часть без коэффициента может лишь не больше 1. Это означает, что наше алгебраическое выражение по смыслу не совпадает с изначальными логическими условиями. Чтобы оно совпадало, нужно разрешить y принимать любые значения при $x \neq 0$. Для этого как раз и добавлен коэффициент в левой части неравенства, чтобы неравенство оставалось верным при $x = 1$ и сколь угодно большой правой части.

3. Рассмотрим самый общий случай: «если некоторые условия из $\mathcal{A}$ истины (а остальные ложны), то некоторые условия из множества $\mathcal{B}$ истины (а остальные ложны)». Для этого введём булевы векторы $x = (x_i)$ и $y = (y_k)$, соответствующие условиям из $\mathcal{A}$ и $\mathcal{B}$ соответственно. Также аналогично с двумя предыдущими пунктами введём множества $I^0, I^1 \subset I$ и $K^0, K^1 \subset K$ следующим образом

$$x_i = \begin{cases} 0, & i \in I^0 \Leftrightarrow A_i \text{ — истина} \\ 1, & i \in I^1 \Leftrightarrow A_i \text{ — ложь} \end{cases}$$

$$y_k = \begin{cases} 0, & k \in K^0 \Leftrightarrow B_k \text{ — истина} \\ 1, & k \in K^1 \Leftrightarrow B_k \text{ — ложь} \end{cases}$$

Фактически нам бы хотелось записать следующее: «если x имеет такой-то вид, то y имеет такой-то вид». Записать алгебраически это можно следующим образом

$$\left|K^0 \cup K^1\right| \cdot \Big(\sum_{i \in I^1} (1 - x_i) + \sum_{i \in I^0} x_i \Big) \geq \sum_{k \in K^1} (1 - y_k) + \sum_{k \in K^0} y_k.$$

Коэффициент в левой части добавлен по аналогии с предыдущим пунктом (иначе при любых значениях вектора x вектор y не может быть любым).

Подытожим

1. «Если x имеет такой-то вид, то $y = 0$»

$$\sum_{i \in I^1} (1 - x_i) + \sum_{i \in I^0} x_i \geq y.$$

2. «Если $x = 0$, то y имеет такой-то вид»

$$\left|K^0 \cup K^1\right| \cdot x \geq \sum_{k \in K^1} (1 - y_k) + \sum_{k \in K^0} y_k.$$

3. «Если x имеет такой-то вид, то y имеет такой-то вид»

$$\left|K^0 \cup K^1\right| \cdot \Big(\sum_{i \in I^1} (1 - x_i) + \sum_{i \in I^0} x_i \Big) \geq \sum_{k \in K^1} (1 - y_k) + \sum_{k \in K^0} y_k.$$

Утверждение 6 (альтернативные переменные)

Пусть в нашей задаче сформулированы 2 ограничения

$$f_1(x) \geq b_1,$$

$$f_2(x) \geq b_2,$$

однако нам достаточно, чтобы выполнялось хотя бы одно из них. Тогда такие условия называются *альтернативными*. Как это можно записать алгебраически?

Предположим, что мы реализуем алгоритм, который решает нашу задачу, при этом он находит решения, которые удовлетворяют всем условиям. Введём булеву переменную y, значения которой определяются следующим образом

$$y = \begin{cases} 0, & \text{выполнено первое ограничение} \\ 1, & \text{выполнено второе ограничение} \end{cases}$$

В идеальном случае наш алгоритм сам выберет значение этой переменной и на его основании построит оптимальное решение, однако какое бы значение он не выбрал, итоговое решение должно удовлетворять одному из ограничений. Алгебраически это можно записать так

$$f_1(x) \geq b_1 - W(1 - y),$$

$$f_2(x) \geq b_2 - Wy,$$

где W — это какая-то большая величина. Какое конкретно значение эта величина принимает, зависит от конкретной задачи. Где-то можно положить $W = 10^6$, где-то $W = 50$, однако полностью избавиться от W и записать условие без него не получится.

- Если $y = 1$, то имеем

$$f_1(x) \geq b_1,$$

$$f_2(x) \geq b_2 - W \to -\infty.$$

Первое неравенство верно, как и второе. Однако если со вторым есть вопросы, а точно ли оно верно, то вот первое выполняется гарантировано.

- Если $y = 0$, то имеем

$$f_1(x) \geq b_1 - W \to -\infty,$$

$$f_2(x) \geq b_2.$$

Аналогично, оба неравенства выполнены, но важно, что второе неравенство выполняется гарантировано.

Другой способ

То же самое можно записать с использованием двух булевых переменных y_1 и y_2

$$f_1(x) \geq b_1 - W(1 - y_1),$$

$$f_2(x) \geq b_2 - W(1 - y_2),$$

$$y_1 + y_2 = 1.$$

Без условия $y_1 + y_2 = 1$ нет гарантий, что одно из неравенств будет верно.

Ограничения в другую сторону

Если у нас есть ограничения с другим знаком

$$f_1(x) \leq b_1,$$

$$f_2(x) \leq b_2,$$

то записать алгебраически их можно следующим образом

$$f_1(x) \leq b_1 + W(1 - y_1),$$

$$f_2(x) \leq b_2 + W y_2,$$

$$y_1 + y_2 = 1.$$

Аналогично можно было бы записать через y.

Утверждение 7 (замена нелинейностей)

Пусть при решение задачи в некотором выражении нам встретилась какая-то нелинейность, например $\hat{x} \cdot \hat{y}$ ($\hat{x}$ и $\hat{y}$ — булевы переменные). Работать с нелинейностями неудобно, поэтому хотелось бы заменить это на новую булеву переменную

$$\hat{z} = \hat{x} \cdot \hat{y}.$$

Однако просто заменить в выражении $\hat{x} \cdot \hat{y}$ на $\hat{z}$ нельзя, поскольку нужно изменить ограничения. Нам нужно ввести ограничение на новую переменную

$$\hat{z} = 1 \quad \Longleftrightarrow \quad \hat{x} = 1 \wedge \hat{y} = 1$$

Данное ограничение можно расписать через две импликации

1. Если $\hat{z} = 1$, то $\hat{x} = 1$ и $\hat{y} = 1$. Данное логическое условие можно записать алгебраически, используя утверждение 5. Будем использовать самый общий случай

$$x = \begin{pmatrix} \hat{z} \end{pmatrix}, \qquad y = \begin{pmatrix} \hat{x} & \hat{y} \end{pmatrix},$$
$$I^0 = \varnothing, \quad I^1 = \{1\}, \qquad K^0 = \varnothing, \quad K^1 = \{1, 2\}$$
$$\left|K^0 \cup K^1\right| \cdot \Big(\sum_{i \in I^1} (1 - x_i) + \sum_{i \in I^0} x_i \Big) \geq \sum_{k \in K^1} (1 - y_k) + \sum_{k \in K^0} y_k,$$
$$\Updownarrow$$
$$2 \cdot (1 - \hat{z}) \geq (1 - \hat{x}) + (1 - \hat{y}),$$
$$2\hat{z} \leq \hat{x} + \hat{y}.$$

2. Если $\hat{x} = 1$ и $\hat{y} = 1$, то $\hat{z} = 1$. Данное логическое условие можно записать алгебраически, используя утверждение 5. Будем использовать самый общий случай

$$x = \begin{pmatrix} \hat{x} & \hat{y} \end{pmatrix}, \qquad y = \begin{pmatrix} \hat{z} \end{pmatrix},$$
$$I^0 = \varnothing, \quad I^1 = \{1, 2\}, \qquad K^0 = \varnothing, \quad K^1 = \{1\}$$
$$\left|K^0 \cup K^1\right| \cdot \Big(\sum_{i \in I^1} (1 - x_i) + \sum_{i \in I^0} x_i \Big) \geq \sum_{k \in K^1} (1 - y_k) + \sum_{k \in K^0} y_k,$$
$$\Updownarrow$$
$$(1 - \hat{x}) + (1 - \hat{y}) \geq 1 - \hat{z},$$
$$\hat{z} + 1 \geq \hat{x} + \hat{y}.$$

Итого при замене $\hat{z} = \hat{x} \cdot \hat{y}$ нужно добавить два ограничения

$$\boxed{\begin{cases} 2\hat{z} \leq \hat{x} + \hat{y}, \\ \hat{z} + 1 \geq \hat{x} + \hat{y}. \end{cases}}$$

1.6.1 Задача о проектах

Пусть есть 5 проектов, в которые можно вложиться. Для вложения в каждый проект нужно внести определённую сумму денег. Все проекты после вложения в них принесут определённый доход через какое-то время. Есть определённые условия, на которых можно вкладываться в проекты. Как получить наибольшую прибыль, имея ограниченные ресурсы?

1. Что будем оптимизировать? Ответ: нужно максимизировать получаемую прибыль.

2. Что существенно влияет на оптимизируемую характеристику?

Проект	A	B	C	D	E
Доход	3	2	1	4	2
Начальные вложения	1.5	0	0.5	4	1

Условия вложения в проекты

1. нужно вложиться хотя бы в один проект;
2. если вложились в A, то необходимо вложиться в D;
3. если вложились в B и C, то необходимо вложиться в A;
4. если вложились в B, то необходимо вложиться в C и E.

Параметры

- c_i — доход с проекта i;
- d_i — начальные вложения в проект i;
- Q — стартовый капитал;
- C — общий доход.

Переменные

- x_1, x_2, x_3, x_4, x_5 — булевы переменные, которые означают, будет ли вложение в соответствующий проект

$$x_i = \begin{cases} 1, & \text{будем вкладываться в } i\text{-ый проект,} \\ 0, & \text{иначе.} \end{cases}$$

3. Математическая формулировка задачи

Пусть D — сумма всех расходов, C — сумма всех доходов, F — прибыль, тогда

$$D = \sum_{i=1}^{5} x_i d_i, \quad C = \sum_{i=1}^{5} x_i c_i,$$

$$F = C - D = \sum_{i=1}^{5} x_i c_i - \sum_{i=1}^{5} x_i d_i = \sum_{i=1}^{5} x_i (c_i - d_i) \to \max_{(x_i)}.$$

Нужно ещё добавить ограничение на то, что мы не можем вложить в проекты больше стартового капитала

$$D \le Q \quad \Longleftrightarrow \quad \sum_{i=1}^{5} x_i d_i \le Q.$$

Условия вложения в проекты

1. Первое условие можно записать алгебраически через

$$\sum_{i=1}^{5} x_i \ge 1$$

2. Второе условие эквивалентно «если $x_1 = 1$, то $x_4 = 1$», алгебраически это записывается так (утверждение 4)

$$1 - x_1 \ge 1 - x_4 \quad \Longleftrightarrow \quad x_1 \le x_4.$$

3. Третье условие эквивалентно «если $x_2 = 1$ и $x_3 = 1$, то $x_1 = 1$», алгебраически это записывается так (утверждение 5)

$$(1 - x_2) + (1 - x_3) \ge 1 - x_1 \quad \Longleftrightarrow \quad 1 + x_1 \ge x_2 + x_3.$$

4. Четвёртое условие эквивалентно «если $x_2 = 1$, то $x_3 = 1$ и $x_5 = 1$», алгебраически это записывается так (утверждение 5)

$$2 \cdot (1 - x_2) \ge 1 - x_3 + 1 - x_5 \quad \Longleftrightarrow \quad 2x_2 \le x_3 + x_5.$$

Итоговая модель

$$\sum_{i=1}^{5} x_i (c_i - d_i) \to \max_{(x_i)}$$

$$\sum_{i=1}^{5} x_i d_i \le Q$$

$$\begin{cases} \sum\limits_{i=1}^{5} x_i \ge 1, \\ x_1 \le x_4, \\ 1 + x_1 \ge x_2 + x_3, \\ 2x_2 \le x_3 + x_5; \end{cases}$$

$$x_i \in \{0, 1\}.$$

1.6.2 Задача о предприятии

Пусть есть предприятие, которое производит определённые виды продукции, затрачивая некоторые свои ресурсы. Для простоты будем считать, что у нас есть лишь один вид ресурсов, который можно использовать для производства продукции. Как получить наибольший доход?

1. Что будем оптимизировать? Ответ: нужно максимизировать получаемый доход.

2. Что существенно влияет на оптимизируемую характеристику?

Пусть $i = 1 \dots n$ — вид продукции, $I = \{1, 2, \dots, n\}$ — список всех производимых товаров.

Параметры

- c_i — доход продукции;
- b_i — расход ресурса на производство одной единицы продукции;
- B — запасы ресурса.

Переменные

- x_i — количество единиц производимой продукции.

3. Математическая формулировка задачи

$$C = \sum_{i \in I} c_i x_i \to \max_{(x_i),}$$

$$\sum_{i \in I} b_i x_i \le B,$$

$$\forall i \in I \quad x_i \ge 0.$$

Изменения в задаче

Казалось бы, математическая модель составлена, однако тут приходит директор предприятия и говорит, что правительство определило два списка социально значимых товаров I_1, I_2, при этом нужно либо из первого списка производить не менее a_1 единиц продукции, либо из второго не менее a_2 единиц продукции. Оба условия можно записать следующим образом

$$\sum_{i \in i_1} x_i \ge a_1, \quad \sum_{i \in I_2} x_i \ge a_2,$$

Нам нужно, чтобы выполнялось хотя бы одно из них. Для этого введём булеву переменную

$$y = \begin{cases} 0, & \text{производим не менее } a_1 \text{ из } I_1, \\ 1, & \text{производим не менее } a_2 \text{ из } I_2; \end{cases}$$

и запишем с её помощью требуемое условие (утверждение 6)

$$\sum_{i \in I_1} x_i \geq a_1 - Wy,$$

$$\sum_{i \in I_2} x_i \geq a_2 - W(1-y).$$

Новые изменения в задаче

Казалось бы, сейчас математическая модель предприятия окончательно составлена, но... К вам вновь приходит директор предприятия и говорит, что правительство издало приказ, по которому если производится достаточно продукции из обоих списков, то предприятие получает надбавку к финансированию.

Пусть если C_0 — надбавка за производство продукции из обоих списков социально значимых товаров. Введём две новые булевы переменные

$$y_1 = \begin{cases} 1, & \text{производим не менее } a_1 \text{ из } I_1, \\ 0, & \text{иначе;} \end{cases}$$

$$y_2 = \begin{cases} 1, & \text{производим не менее } a_2 \text{ из } I_2, \\ 0, & \text{иначе.} \end{cases}$$

Изменим выражение для оптимизируемой характеристики и логических условий

$$C = \sum_{i \in I} c_i x_i + C_0 y_1 y_2 \to \max_{(x_i), y_1, y_2}$$

$$\sum_{i \in I_1} x_i \geq a_1 - W(1-y_1),$$

$$\sum_{i \in I_2} x_i \geq a_2 - W(1-y_2).$$

Замена нелинейности

В общем и целом теперь нас всё устраивает, кроме нелинейности в виде $y_1 y_2$. Произведём замену нелинейности и запишем ограничения на новую булеву переменную (утверждение 7)

$$z = y_1 y_2,$$

$$C = \sum_{i \in I} c_i x_i + C_0 z \to \max_{(x_i), z},$$

$$2z \le y_1 + y_2,$$

$$z + 1 \ge y_1 + y_2.$$

Итоговая модель

Параметры: $\{c_i\}$, $\{b_i\}$, B, a_1, a_2, C_0.

Переменные: $\{x_i\}$, y_1, y_2, z.

$$C = \sum_{i \in I} c_i x_i + C_0 z \to \max_{(x_i), z},$$

$$\sum_{i \in I} b_i x_i \le B,$$

$$\sum_{i \in I_1} x_i \ge a_1 - W(1 - y_1),$$

$$\sum_{i \in I_2} x_i \ge a_2 - W(1 - y_2),$$

$$2z \le y_1 + y_2, \quad z + 1 \ge y_1 + y_2,$$

$$y_1 \in \{0, 1\} \quad y_2 \in \{0, 1\}, \quad z \in \{0, 1\},$$

$$\forall i \in I \quad x_i \ge 0.$$

Глава 2

Динамическое программирование

Динамическое программирование эффективно, если процесс принятия решения состоит из многих шагов, то есть когда итоговое решение — последовательность принимаемых решений (*стратегия*). Теоретически с помощью динамического программирования можно решить любую экстремальную задачу, однако практически это не всегда возможно, так как появляется слишком много состояний... **Главная идея динамического программирования** заключается в том, чтобы не пытаться решить задачу непосредственно, а поместить её в семейство аналогичных задач, среди которых есть просто решаемые.

2.1 Задача загрузки судна

Пример

Есть грузовое судно и набор различных контейнеров. Требуется загрузить судно контейнерами таким образом, чтобы при их дальнейшей продаже заработать как можно больше.

Формальное описание:

- есть N контейнеров;
- x_i — сколько контейнеров с номером i нужно взять на судно;
- h_i — место на грузовой площадке, занимаемое контейнером с номером i;
- c_i — ценность контейнера с номером i;
- P — размер грузовой площадки судна;

$$\sum_{i=1}^{N} c_i x_i \to \max_{(x_i),\, i\in\{1,\dots,N\}}$$

$$\sum_{i=1}^{N} h_i x_i \le P$$

$$x_i \in \{0,1,2,3,\dots\}, \quad i \in 1,\dots N$$

Решение

Для решения задачи рассмотрим семейство аналогичных задач. Задачи данного семейства будут охарактеризовываться парой (n,p), где

- p — место на площадке;
- n — минимальный номер для контейнеров, которые у нас есть.

$$n = 1 \dots N \qquad p = 0 \dots P.$$

Вместо рассмотрения общей задачи про все N контейнеров и всю грузовую площадку размером P будем рассматривать задачи, в которых нам «доступны» не все контейнеры, а лишь начинающиеся с номера n, а также в которых нам «недоступна» вся грузовая площадка судна, а лишь её часть размером p.

Запишем целевую функцию и ограничения для элемента семейства

$$\sum_{i=n}^{N} c_i x_i \to \max_{(x_i),\, i\in\{n,\dots N\}}$$

$$\sum_{i=n}^{N} h_i x_i \le p$$

$$x_i \in \{0,1,2,3,\dots\} \quad i \in \{n,\dots,N\}$$

Мы формально записали формулировку задачи для некоторого элемента семейства, при этом этот элемент характеризуется парой (n,p).

Зададимся вопросом: а есть ли в этом семействе задачи, которые можно легко решить? Да, например если $n = N$, то есть если у нас в распоряжении есть лишь контейнеры с номером N.

Введём обозначение. Пусть $\boxed{f_n(p)}$ — оптимальное значение целевой функции задачи (n,p). По своей суть $f_n(p)$ —это максимальный доход, который мы получим, если будем на площадку размера p грузить контейнеры с номерами $n, n+1, n+2, \dots, N$.

Легко заметить, что

$$f_N(p) = \left[\frac{p}{h_N}\right],$$

то есть решать задачу (N, p) для любого $p \leq P$ мы умеем. Теперь нужно совершить переход к решению исходной задачи

$$f_N(p) \longrightarrow f_1(P).$$

Для этого сформулируем принцип.

Определение (принцип оптимальности для оптимальной стратегии)

Оптимальная стратегия обладает тем свойством, что каким бы не было первое решение, последующие решения должны образовывать оптимальную стратегию относительно состояния, полученного по итогам первого решения.

Пример

Пусть мы стоим у доски и нам захотелось как можно быстрее выйти из аудитории через дверь. Каким бы ни был наш первый шаг, если мы хотим дойти до двери как можно быстрее, придётся всё время действовать оптимально. То есть даже если первый шаг был оптимальным, но потом мы накосячили и пошли неоптимально, стратегия точно не получится оптимальной.

Вернёмся к задаче. Предположим, что мы умеем считать

$$f_{n+1}(p), f_{n+2}(p), \ldots, f_N(p),$$

однако нам бы хотелось посчитать $f_n(p)$, как это можно сделать? Предположим, что $x_n = x = 1$, тогда груз с номером n даст нам ценность $c_n \cdot x = c_n$ и займёт на площадке место $h_n \cdot x = h_n$. Запишем целевую функцию

$$\forall p \leq P \qquad f_n(p) = \max_{h_n \cdot x \leq p,\ x=0,1,2,\ldots} \big\{c_n \cdot x + f_{n+1}(p - h_n \cdot x)\big\}.$$

Мы получили **рекуррентное соотношение динамического программирования**. На данном этапе $f_n(p)$ можно получить перебором значение $x_n = x$, а $f_{n+1}(\ldots)$ нам уже известно.

Какой здесь будет алгоритм при реализации? Алгоритм будет включать в себя N шагов, на каждом из которых мы будем вычислять $f_n(p)$. По сути мы будем заполнять табличку

Для решения задачи нам важно лишь $f_1(P)$, считать $f_1(p)$ для всех $p \leq P$ не нужно.

Алгоритм имеет две стадии:

1. *обратный ход* $N \to 1$ — заполнение таблицы,
2. *прямой ход* $1 \to N$ — вычисление оптимального решения.

p	$f_1(p)$	$f_2(p)$	$\dots$	$f_n(p)$	$f_{N-1}(p)$	$f_N(p)$
0	$\times$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
1	$\times$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
2	$\times$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
3	$\times$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
$P-1$	$\times$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
P	$\dots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

Сначала мы будем идти с конца и для всех значений p считать $f_N(p)$. Потом на основании этого будем для всех значений p считать $f_{N-1}(p)$ и так далее. Когда мы заполним все столбцы, кроме $f_1(p)$, мы посчитаем $f_1(P)$ и на этом заполнение таблички закончим.

Как теперь по табличке получить само оптимальное решение — отдельный вопрос, который будет рассмотрен позже.

2.2 N-шаговый процесс принятия решений

Пусть N — число шагов в нашем процессе. Исходные данные задачи:

- P_i — множество возможных состояний на i-ом шаге, $i = 1 \dots N+1$. Шага с номером $N+1$ не будет, поэтому p_{N+1} можно считать фиктивной переменной, введённой для удобства. В рассмотренной ранее задачи множеством наших состояний были числа $\{0, 1, 2, \dots, P\}$ — то есть размер на площадке, занимаемым всеми выбранными контейнерами на некотором шаге;

- $p_1 = p_0$ — некоторая известная заданная величина;

- Q_i — множество возможных решений на i-ом шаге, $i = 1 \dots N$.

- $T_i(p, q)$ — функции перехода из состояния p в состояние q на i-ом шаге, $i = 1 \dots N$;

- $Q_i(p)$ — множество допустимых решений на i-ом шаге в состоянии p, то есть

$$Q_i(p) = \{q \in Q_i \qquad T_i(p, q) \in P_{i+1}\}$$

Берём $q \in Q_1(p_0)$. На каждом шаге выбираем состояние из множества допустимых состояний. То есть $\{q_1, q_2, \ldots, q_n\}$ — наша допустимая стратегия. С помощью функции перехода переводят нас в допустимое решение, то есть

$$p_{i+1} = T_i(p_i, q_i) \in P_{i+1}$$

Как понять, что стратегия хорошая? Введём $g_i(p, q)$ — функция дохода на i-ом шаге, если мы в состоянии p принимаем решение q.

2.3 Задача выбора оптимальной стратегии N-шагового процесса

Пусть у нас задача с теми же исходными данными:

- N — число шагов;
- $g_i(p, q)$ — функция дохода на i-ом шаге, если мы в состоянии p принимаем решение q;
- P_i — ...;

...

Нам нужно:

$$\max_{\{q_1, q_2, \ldots, q_N\}} \sum_{i=1}^{N} g_i(p_i, q_i)$$

с ограничениями

$$p_1 = p_0 \qquad p_i \in P_i, i \in \{2, \ldots, N+1\}$$

$$q_i \in Q_i, i \in \{1, \ldots, N\}$$

$$p_{i+1} = T_i(p_i, q_i), i \in \{1, \ldots, N\}$$

Решим задачу по аналогии с задачей погрузки судна: рассмотрим семейство задач и рассмотрим элемент семейства (n, p). Рассмотрим данный элемент семейства (шаг n)

$$\max_{\{q_n, q_{n+1}, \ldots, q_N\}} \sum_{i=n}^{N} g_i(p_i, q_i)$$

с ограничениями

$$p_n = p$$

$$p_i \in P_i, i \in \{n+1, \ldots, N+1\}$$

$$q_i \in Q_i, i \in \{n, \ldots, N\}$$

$$p_{i+1} = T_i(p_i, q_i), i \in \{n, \ldots, N\}$$

В данных обозначениях исходная задача — это $f_1(p_0)$. Напишем рекуррентные соотношения в общем виде. Пускай мы знаем как решать «простую задачу»

$$\forall p \in P_N \qquad f_N(p) = \max_{q \in Q_N(p)} g_N(p, q)$$

Теперь нам нужно осуществить переход от «простой задачи» к исходной. Пусть мы находимся на шаге n, тогда

$$f_n(p) = \max_{q \in Q_n(p)} \Big[g_n(p, q) + \underbrace{f_{n+1}\big(T_n(p, q)\big)}_{\text{уже знаем решение}} \Big]$$

Но мы не знаем, как выглядят оптимальные решения на каждом шаге, то мы не знаем стратегию. Для этого будем на каждом шаге считать $f_n(p)$ и $q_n(p)$ — значение q, на котором достигается максимум в состоянии p. Это значение запоминаем.

$$p_1^* = p_0, \qquad q_1^* = q_1(p_1^*),$$

$$p_2^* = T_1(p_1^*, q_1^*), \qquad q_2^* = q_2(p_2^*),$$

$$p_3^* = T_2(p_2^*, q_2^*), \qquad q_3^* = q_3(p_3^*),$$

$$\ldots$$

$$p_N^* = T_{N_1}(p_{N_1}^*, q_{N_1}^*), \qquad q_N^* = q_N(p_N^*),$$

Совокупность всех значений $\{q_i^*\}_{i=1}^N$ и будет нашей оптимальной стратегией. Однако почему это будет именно оптимальной стратегией?

Итак, имеем стратегию $\{q_1^*, q_2^*, \ldots, q_N^*\}$ — стратегия. Докажем, что она оптимальна. Покажем, что

$$f_1(p_0) = \sum_{i=1}^{N} g_i(p_i^*, q_i^*).$$

Доказательство

$$f_1(p_0) \stackrel{def}{=} f_1(p_1^*) = \max_{q \in Q_1(p_1^*)} \Big[g_1(p_1^*, q) + f_2\big(T_1(p_1^*, q)\big) \Big]$$

По определению максимум происходит при $q = q_1^* = q(p_1^*)$, поэтому

$f_1(p_0) = g_1(p_1^*, q_1^*) + f_2\big(T_1(p_1^*, q_1^*)\big) = g_1(p_1^*, q_1^*) + g_2(p_2^*, q_2^*) + \cdots = \cdots =$ имеем, что нужно

Пример (о машине на острове)

На некотором острове есть машина; для работы машины нужны детали, которые могут изнашиваться. В скором времени нужно будет вызвать самолёт, который сможет доставить необходимые детали, однако максимальная масса груза ограничена. Сколько деталей какого типа нужно заказать?

1. Что будем оптимизировать? Ответ: машина должна работать максимальное время.

2. Что существенно влияет на оптимизируемую характеристику?

- $i = 1 \dots N$ — вид детали;
- $h = \{h_i\}_{i=1}^N$ — масса деталей;
- $t = \{t_i\}_{i=1}^N$ — срок службы деталей;
- $x = \{x_i\}_{i=1}^N$ — сколько нужно взять деталей в посылку;
- $d = \{d_i\}_{i=1}^N$ — количество работающих деталей в машине;
- P — максимальная масса посылки;

3. Математическая формулировка задачи

Пусть T — время работы машины, тогда

$$T = \min_{i=1 \dots N} \big((x_i + d_i) \cdot t_i \big) \to \max$$

$$\sum_{i=1}^{N} h_i x_i \le P$$

Решение

Решим задачу с помощью динамического программирования следующим образом

- количество шагов процесса = количество видов деталей $= N$;

- на i-ом шаге будем определять, сколько деталей i-го типа нужно взять в посылку;
- текущее состояние — свободная масса груза в посылке, то есть сколько ещё массы можно использовать.

Будем рассматривать семейства задач, которые определяются парой (n, p). Фактически это означает, что у нас в распоряжении

- есть детали не всех видов, а лишь от n до N, $n \leq N$;
- есть не вся масса посылки P, а лишь её часть $p \leq P$.

Пусть $f_n(p)$ — максимальная время, которое проработает машина в семействе задаче (n, p). Заметим, что если $n = N$, то задача легко решается

$$\forall p \leq P \qquad f_N(p) = \left(d_N + \left[\frac{p}{h_n}\right]\right) \cdot t_N$$

Тогда решение состоит в том, что нужно заказать деталей вида N на максимум, то есть сколько можем, столько и заказываем.

Теперь когда у нас база для решения задачи, осуществим переход $f_{n+1} \to f_n(p)$ в предположении, что $f_{n+1}(p)$ мы знаем $\forall p \leq P$.

$$\boxed{f_n(p) = \max_{\substack{x=x_n \\ h_n \cdot x \leq p,\ x \geq 0}} \min\Big\{(d_n + x) \cdot t_n\ ;\ f_{n+1}(p - x \cdot h_n)\Big\}}$$

$(d_n + x) \cdot t_n$ — это сколько проработают детали n-го вида с учётом уже имеющихся в машине и тех, которые будут заказаны ($x = x_n$ — сколько деталей данного вида нужно заказать), а $f_{n+1}(p - x \cdot h_n$ — это по предположению уже известное максимальное время работы всех остальных деталей. Для решения задачи нужно для всех $p \leq P$ найти значение $f_n(p)$ и соответствующие значения x, на которых максимум и достигается.

В исходной формулировке у нас было

$$f_n(p) = \max_{q \in Q_n(p)} \Big[g_n(p, q) + f_{n+1}\big(T_n(p, q)\big)\Big],$$

Отличие в задаче про машину состоит в том, что у нас в рекуррентном соотношении не суммирование, а взятие минимума. Теоретически здесь может быть и умножение, но большой роли при решении это не играет. Важно, что написано рекуррентное соотношение.

Конкретная задача

- $N = 4$;
- $h = \{3, 2, 3, 2\}$ (кг);

- $t = \{6, 3, 2, 4\}$ (дней);
- $d = \{1.5, 1.5, 1.5, 1.5\}$.

$d_i = 1.5$ может означать, например, что в машине установлены две детали вида i, при этом одна отработала половину своего срока, а вторую только недавно установили.

Помним, что наши состояния — это оставшаяся масса груза p.

p	(f_1, q_1)	(f_2, q_2)	(f_3, q_3)	(f_4, q_4)
0	$\times$	$(3, 0)$	$(3, 0)$	$(6, 0)$
1	$\times$	$(3, 0)$	$(3, 0)$	$(6, 0)$
2	$\times$	$(3, 0/1)$	$(3, 0)$	$(10, 1)$
3	$\times$	$(4.5, 0)$	$(5, 1)$	$(10, 1)$
4	$\times$	$(4.5, 0)$	$(5, 1)$	$(14, 2)$
5	$\times$	$(5, 0)$	$(5, 1)$	$(14, 2)$
6	$\times$	$(5, 1)$	$(6, 2)$	$(18, 3)$
7	$\times$	$\times$	$\times$	$\times$
8	$(6, 0)$	$(6, 1)$	$(7, 2)$	$(22, 4)$

Будем заполнять таблицу справа налево, запоминая q_i — то значение x, на котором достигается максимум.

$\boxed{n = 4}$ На первом шаге $n = N = 4$, запишем выражение для $f_4(p)$

$$f_4(p) = \left(d_n + \left[\frac{p}{h_4}\right]\right) \cdot t_4 = \left(1.5 + \left[\frac{p}{2}\right]\right) \cdot 4,$$

$$q_4 = \left[\frac{p}{2}\right].$$

То есть мы можем посчитать значение $f_4(p)$ для любого $p \leq P$, при этом нам известно, на котором достигается максимум (q_4).

Посчитаем значение f_4 для всех возможных состояний

$$f_4(0) = 6, \quad q_4 = \left[\frac{0}{2}\right] = 0;$$

$$f_4(1) = 6, \quad q_4 = \left[\frac{0}{2}\right] = 0;$$

$$f_4(2) = 10, \quad q_4 = \left[\frac{0}{2}\right] = 1;$$

$$f_4(3) = 10 \quad q_4 = \left[\frac{0}{2}\right] = 1;$$

$$f_4(4) = 14, \quad q_4 = \left[\frac{0}{2}\right] = 2;$$

$$f_4(5) = 14, \quad q_4 = \left[\frac{0}{2}\right] = 2;$$

$$f_4(6) = 18, \quad q_4 = \left[\frac{0}{2}\right] = 3;$$

$$f_4(8) = 22, \quad q_4 = \left[\frac{0}{2}\right] = 4.$$

Занесём все эти данные в последний столбец таблицы.

$\boxed{n = 3}$ На втором шаге $n = 3$. Запишем выражение для $f_3(p)$

$$f_3(p) = \max_{\substack{x=x_3 \\ h_3 \cdot x \le p,\ x \ge 0}} \min\Big\{(d_3 + x) \cdot t_3\ ;\ f_4(p - x \cdot h_3)\Big\}$$
$$= \max_{\substack{x=x_3 \\ 3x \le p,\ x \ge 0}} \min\Big\{3 + 2x\ ;\ f_4(p - 3x)\Big\}$$

На данном шаге нам нужно для всех $0 \le p \le P$ вычислить значение $f_3(p)$ и запомнить значение x, на котором достигается максимум в каждом конкретном состоянии. Для каждого состояния мы будем перебирать различные значения x. Для примера найдём $f_3(8)$. Для этого нам нужно перебрать значения $x \in \{0, 1, 2\}$. При $x > 2$ неравенство $3x \le 8$ уже не выполняется, а $x < 0$ нам не подходят по условию. Вычислим значение $\min$ для каждого из этих трёх значений x

$$x = 0: \qquad \min\big\{3 + 2 \cdot 0\ ;\ f_4(8)\big\} = \min\{3, 22\} = 3$$
$$x = 1: \qquad \min\big\{3 + 2 \cdot 1\ ;\ f_4(5)\big\} = \min\{5, 14\} = 5$$
$$x = 2: \qquad \min\big\{3 + 2 \cdot 2\ ;\ f_4(2)\big\} = \min\{7, 10\} = \textcircled{7}$$

То есть мы рассмотрели три разных значения x (0, 1, 2), для каждого из них вычислили значение $\min\Big\{(d_n + x) \cdot t_n\ ;\ f_{n+1}(p - x \cdot h_n)\Big\}$, а после этого выбрали из трёх итоговых значение максимальное — 7, при этом запомнили, что это значение достигается при $x = 2$. Запишем это в таблицу

Однако данные вычисления можно записать существенно короче

$$f_3(8) = \begin{array}{r|l} 0 & \{3\ ;\ f_4(8) = 22\} = 3 \\ 1 & \{5\ ;\ f_4(8) = 14\} = 5 \\ 2 & \{7\ ;\ f_4(8) = 10\} = \textcircled{7} \end{array}$$

В первом столбце у нас идут перебираемые значения x, а далее для каждого из них вычисление $\min$, само слово «min» писать здесь излишне. В кружок обведено значение $\max$. Данной нотации и будем придерживаться всюду далее. Посчитаем f_3 для всех p

$$f_3(0) = \begin{array}{r|l} 0 & \{3\ ;\ f_4(0) = 6\} = \textcircled{3} \end{array}$$

$$f_3(1) = \begin{array}{r|l} 0 & \{3\ ;\ f_4(1) = 6\} = \textcircled{3} \end{array}$$

$$f_3(2) = \begin{array}{r|l} 0 & \{3\ ;\ f_4(2) = 10\} = \textcircled{3} \end{array}$$

$$f_3(3) = \begin{array}{r|l} 0 & \{3\ ;\ f_4(3) = 10\} = 3 \\ 1 & \{5\ ;\ f_4(0) = 6\} = \textcircled{5} \end{array}$$

$$f_3(4) = \begin{array}{r|l} 0 & \{3\ ;\ f_4(4) = 14\} = 3 \\ 1 & \{5\ ;\ f_4(1) = 6\} = \textcircled{5} \end{array}$$

$$f_3(5) = \begin{array}{r|l} 0 & \{3\ ;\ f_4(5) = 14\} = 3 \\ 1 & \{5\ ;\ f_4(2) = 10\} = \textcircled{5} \end{array}$$

$$f_3(6) = \begin{array}{r|l} 0 & \{3\ ;\ f_4(6) = 18\} = 3 \\ 1 & \{5\ ;\ f_4(3) = 10\} = 5 \\ 2 & \{7\ ;\ f_4(0) = 6\} = \textcircled{6} \end{array}$$

$$f_3(8) = \begin{array}{r|l} 0 & \{3\ ;\ f_4(8) = 22\} = 3 \\ 1 & \{5\ ;\ f_4(3) = 10\} = 5 \\ 2 & \{7\ ;\ f_4(2) = 10\} = \textcircled{7} \end{array}$$

Занесём все данные в таблицу. Ещё раз повторение: В столбец (f_3, q_3) возле максимального значения, обведённого в кружочек, для каждого состояния p мы ещё записываем значение x, в котором достигается максимум. Так, для $f_3(8)$ это $x = 2$, для $f_3(5)$ это $x = 1$ и т.д.

$\boxed{n=2}$ На третьем шаге $n=2$

$$f_2(p) = \max_{\substack{x=x_2 \\ h_2 \cdot x \le p,\ x \ge 0}} \min\Big\{(d_2+x)\cdot t_2\ ;\ f_3(p - x\cdot h_2)\Big\}$$
$$= \max_{\substack{x=x_2 \\ 2x \le p,\ x \ge 0}} \min\Big\{4.5 + 3x\ ;\ f_3(p-2x)\Big\}$$

$$f_2(0) = \begin{array}{c|l} 0 & \{4.5\ ;\ f_3(0)=3\} = \textcircled{3} \end{array}$$

$$f_2(1) = \begin{array}{c|l} 0 & \{4.5\ ;\ f_3(1)=3\} = \textcircled{3} \end{array}$$

$$f_2(2) = \begin{array}{c|l} 0 & \{4.5\ ;\ f_3(2)=3\} = \textcircled{3} \\ 1 & \{7.5\ ;\ f_3(0)=3\} = \textcircled{3} \end{array}$$

$$f_2(3) = \begin{array}{c|l} 0 & \{4.5\ ;\ f_3(3)=5\} = \textcircled{4.5} \\ 1 & \{7.5\ ;\ f_3(1)=3\} = 3 \end{array}$$

$$f_2(4) = \begin{array}{c|l} 0 & \{4.5\ ;\ f_3(4)=5\} = \textcircled{4.5} \\ 1 & \{7.5\ ;\ f_3(2)=3\} = 3 \\ 2 & \{10.5\ ;\ f_3(0)=3\} = 3 \end{array}$$

$$f_2(5) = \begin{array}{c|l} 0 & \{4.5\ ;\ f_3(5)=5\} = 4.5 \\ 1 & \{7.5\ ;\ f_3(3)=5\} = \textcircled{5} \\ 2 & \{10.5\ ;\ f_3(1)=3\} = 3 \end{array}$$

$$f_2(6) = \begin{array}{c|l} 0 & \{4.5\ ;\ f_3(6)=6\} = 4.5 \\ 1 & \{7.5\ ;\ f_3(4)=5\} = \textcircled{5} \\ 2 & \{10.5\ ;\ f_3(2)=3\} = 3 \\ 3 & \{13.5\ ;\ f_3(0)=3\} = 3 \end{array}$$

$$f_2(8) = \begin{array}{c|l} 0 & \{4.5\ ;\ f_3(8)=7\} = 4.5 \\ 1 & \{7.5\ ;\ f_3(6)=6\} = \textcircled{6} \\ 2 & \{10.5\ ;\ f_3(4)=5\} = 5 \\ 3 & \{13.5\ ;\ f_3(2)=3\} = 3 \\ 4 & \{16.5\ ;\ f_3(0)=3\} = 3 \end{array}$$

$\boxed{n=1}$ На четвёртом шаге $n = 1$

Наша исходная задача — это $f_1(8)$. Для всех остальных значений $p \neq 8$ $f_1(p)$ не нужно считать.

$$1_3(8) = \max_{\substack{x=x_1 \\ h_1 \cdot x \leq 8,\ x \geq 0}} \min\Big\{(d_1 + x) \cdot t_1\ ;\ f_2(8 - x \cdot h_1)\Big\}$$
$$= \max_{\substack{x=x_1 \\ 3x \leq 8,\ x \geq 0}} \min\Big\{9 + 6x\ ;\ f_2(8 - 3x)\Big\}$$

$$f_1(8) = \begin{array}{r|l} 0 & \{9\ ;\ f_2(8) = 6\} = \textcircled{6} \\ 1 & \{15\ ;\ f_2(5) = 5\} = 5 \\ 2 & \{21\ ;\ f_2(2) = 3\} = 5 \end{array}$$

Вспомним, что $f_1(8)$ — максимальное время работы машины для исходной задачи. Поскольку мы получили, что $f_1(8) = 6$, то в исходной задаче после заказа груза машина проработает ещё 6 дней.

Как же теперь загрузить самолёт?

(a) $p_1^* = 8$, $q_1^* = 0$

(b) $p_2^* = 8$, $q_2^* = 1$

(c) $p_3^* = 8 - h_2 \cdot q_2^* = 6$, $q_3^* = 2$

(d) $p_4^* = p_3^* - h_3 \cdot q_3^* = 0$, $q_4^* = 0$

То есть оптимальное решение — это

$$x_1 = q_1^* = 0,$$
$$x_2 = q_2^* = 1,$$
$$x_3 = q_3^* = 2,$$
$$x_4 = q_4^* = 0.$$

То есть наша стратегия — это $\{0, 1, 2, 0\}$; при данной стратегии машина проработает ещё 6 дней. При любых других стратегиях время работы будет меньше.

Замечание

Чтобы не заполнять всю табличку, можно было в начале прибегнуть к оптимизации, посчитав множество возможных состояний для каждого шага. Тогда бы

мы считали на шаге i значения $f_i(p)$ не для всех $p \leq P$, а лишь для этих самых возможных состояний (значений).

Например, можно заметить, что для подсчёта $f_1(8)$ нам нужно было знать лишь $f_2(8)$, $f_2(5)$ и $f_2(2)$. Значение $f_2(p)$ для остальных p нам в итоге вообще не пригодились.