

## 2. Динамическое программирование

В основе метода лежит идея рассмотрения исходной задачи как представителя семейства сходных с ней задач. Динамическое программирование (ДП) связано с многошаговым (многоэтапным) процессом принятия решений. При этом под многошаговым процессом принятия решений понимается деятельность, при которой принимаются последовательные решения, направленные на достижение одной цели. Методу ДП посвящено множество публикаций, в частности, работы [1,2,5], в которых достаточно подробно рассмотрена техника решения задач методом динамического программирования.

Мы в данном разделе будем изучать метод постепенно по принципу «от простого к сложному». Начнем с распределительной задачи.

### 2.1. Распределительная задача

Сформулируем распределительную задачу на примере планирования деятельности предприятия на  $n$  лет. Пусть предприятие имеет ресурс в размере  $Y$  единиц, который оно может вложить в производство в течение  $n$  лет. Функции  $f_t(x)$  отражают эффективность использования  $x$  единиц ресурса в год  $t$ . Требуется определить план расхода имеющегося ресурса по годам, чтобы максимизировать суммарную эффективность.

Обозначим через  $x_t$  искомую величину ресурса, вкладываемого в развитие производства в год  $t=1, \dots, n$ . Тогда математическая модель может быть записана в виде

$$\sum_{t=1}^n f_t(x_t) \rightarrow \max_{\{x_t\}}; \quad (2.1)$$

$$\sum_{t=1}^n x_t \leq Y; \quad (2.2)$$

$$x_t \geq 0, \quad t=1, \dots, n. \quad (2.3)$$

Решение задачи (2.1)-(2.3) будем строить шаг за шагом, оптимизируя на текущем шаге размер инвестиций в год  $t$ . Будем предполагать функции  $f_t(x)$  неубывающими, что позволяет перейти от исходной задачи к равносильной задаче (2.1), (2.2'), (2.3), в которой неравенство (2.2) заменено равенством

$$\sum_{t=1}^n x_t = Y. \quad (2.2')$$

Используем для задачи (2.1), (2.2'), (2.3) обозначение  $\langle n, Y \rangle$ . Кроме того, обозначим:

- $S^*$  – оптимальное значение целевой функции (2.1);
- $x^* = (x_1^*, \dots, x_n^*)$  – оптимальное решение рассматриваемой задачи, т.е.

$$S^* = \sum_{k=1}^n f_k(x_k^*);$$

- $y_k^* = \sum_{i=1}^k x_i^*$  – оптимальное вложение ресурса за  $k$  первых лет,  $k=1, \dots, n$ .

Далее воспользуемся терминологией и обозначениями из [2]. Наряду с исходной задачей  $\langle n, Y \rangle$  рассмотрим семейство задач  $\pi = \{ \langle k, y \rangle : k=1, \dots, n; 0 \leq y \leq Y \}$ . Пусть  $S_k(y)$  – оптимальное значение целевой функции задачи  $\langle k, y \rangle$ , тогда  $S^* = S_n(Y)$ .

**Теорема 2.1.** Для задачи (2.1), (2.2'), (2.3) справедливы следующие рекуррентные соотношения ДП:

$$S_1(y) = f_1(y), \quad 0 \leq y \leq Y; \quad (2.4)$$

$$S_k(y) = \max_{0 \leq x \leq y} [S_{k-1}(y-x) + f_k(x)], \quad k=2, \dots, n; \quad 0 \leq y \leq Y. \quad (2.5)$$

Значение переменной  $x$ , при котором достигается максимум в (2.5) обозначим через  $x_k(y)$  и назовем *условно-оптимальным решением*.

**Следствие 2.1.** Условно-оптимальное решение  $x_k(y_k^*)$  является оптимальным значением  $k$ -ой компоненты вектора  $x^*$  исходной задачи  $\langle n, Y \rangle$ , т.е.  $x_k^* = x_k(y_k^*)$ ,  $k=1, \dots, n$ .

Алгоритм ДП состоит из *прямого хода* (процесса последовательного вычисления величин  $S_k(y)$ ,  $k=1, \dots, n$ ;  $0 \leq y \leq Y$ ) и *обратного хода* (восстановления оптимального решения). На последнем шаге прямого хода получаем оптимальное значение последней переменной  $x_n^* = x_n(Y)$ . Пусть уже найдены оптимальные значения  $x_n^*, \dots, x_{k+1}^*$ . Тогда  $x_k^* = x_k(y_k^*)$ , где  $y_k^* = y_{k+1}^* - x_{k+1}^*$ .

Схема (2.4), (2.5), как правило, требует численного расчета, но иногда удается получить выражение функций  $S_k(y)$  в аналитическом виде.

Пример 2.1. Рассмотрим метод ДП для задачи (2.1)-(2.3) с функциями  $f_i(x_i) = \frac{x_i^2}{c_i}$ , где  $c_i > 0$  и  $i=1, \dots, n$ .

Прямой ход. На первом шаге получаем  $S_1(y) = \frac{y^2}{c_1}$  и условно-оптимальное решение  $x_1(y) = y$ . Далее

$$S_2(y) = \min_{0 \leq x \leq y} \{f_2(x) + S_1(y-x)\} = \min_{0 \leq x \leq y} \left\{ \frac{x^2}{c_2} + \frac{(y-x)^2}{c_1} \right\}.$$

Выражение в фигурных скобках представляет выпуклую функцию, минимум которой можно найти, приравняв нулю ее производную.

Получим  $\frac{x}{c_2} - \frac{y-x}{c_1} = 0$ , откуда  $x_2(y) = \frac{c_2 y}{c_1 + c_2}$ , а  $S_2(y) = \frac{y^2}{c_1 + c_2}$ .

С помощью математической индукции нетрудно доказать, что

$$S_k(y) = \frac{y^2}{\sum_{i=1}^k c_i}, \quad x_k(y) = \frac{c_k y}{\sum_{i=1}^k c_i}, \quad 0 \leq y \leq Y.$$

Таким образом

$$S^* = S_n(Y) = \frac{Y^2}{\sum_{i=1}^n c_i}, \quad x_n^* = x_n(Y) = \frac{c_n Y}{\sum_{i=1}^n c_i}.$$

Обратный ход. Зная оптимальное значение последней переменной  $x_n^*$ , находим

$$y_{n-1}^* = Y - x_n^* = Y - \frac{c_n Y}{\sum_{i=1}^n c_i} = \frac{Y \sum_{i=1}^{n-1} c_i - c_n Y}{\sum_{i=1}^n c_i} = \frac{Y \sum_{i=1}^{n-1} c_i}{\sum_{i=1}^n c_i}.$$

Следовательно,

$$x_{n-1}^* = x_{n-1}(y_{n-1}^*) = \frac{c_{n-1} Y}{\sum_{i=1}^{n-1} c_i}.$$

Воспользовавшись математической индукцией, получим

$$x_k^* = \frac{c_k Y}{\sum_{i=1}^n c_i}, \quad k=1, \dots, n.$$

В общем случае для реализации алгоритма ДП нужна дискретность значений  $x_k$ . Пусть переменные  $x_k$  целые. Тогда величины  $y$  также целые. Следовательно, параметру  $y$  достаточно принимать значения из конечного множества  $\{0, 1, \dots, Y\}$ . Трудоемкость (сложность) алгоритма в этом случае равна  $O(nY^2)$ , а требуемая память –  $O(nY)$ .

Пример 2.2. На железнодорожную станцию прибыло 8 контейнеров, которые необходимо развезти по 5 складам. Емкость  $i$ -го склада –  $v_i$  контейнеров, затраты на транспортировку одного контейнера на этот склад –  $g_i$ , а стоимость хранения  $x$  контейнеров –  $c_i(x)$ . Требуется развезти все прибывшие контейнеры по складам, чтобы суммарные затраты на транспортировку и хранение были минимальны.

Исходные данные задачи приведены в таблицах 2.1 и 2.2.

	Склады				
	1	2	3	4	5
$g_i$	0.5	1	1.2	1.5	2
$v_i$	2	3	3	5	5

Таб. 2.1.

$x$	$c_1(x)$	$c_2(x)$	$c_3(x)$	$c_4(x)$	$c_5(x)$
1	2	1.5	1	0.5	0.3
2	4	2	2	1	0.5
3	-	3	3	1.5	1
4	-	-	-	2	1.5
5	-	-	-	2.5	2

Таб. 2.2.

Решение. Для записи математической постановки задачи введем функции  $h_i(x) = g_i x + c_i(x)$ ,  $i=1, \dots, 5$ , которые задаются таблицей 2.3. Тогда математическая модель имеет следующий вид.

$$\sum_{i=1}^5 h_i(x_i) \rightarrow \min_{x_i \in \{0, 1, \dots, v_i\}} ;$$

$$\sum_{i=1}^5 x_i = 8.$$

$x$	$h_1(x)$	$h_2(x)$	$h_3(x)$	$h_4(x)$	$h_5(x)$
1	2.5	2.5	2.2	2	2.3
2	5	4	4.4	4	4.5
3	-	6	6.6	6	7
4	-	-	-	8	9.5
5	-	-	-	10	12

Таб. 2.3.

Это распределительная задача и для нее справедливы рекуррентные соотношения:

$$S_l(y) = h_l(y), y=0, 1, \dots, 8;$$

$$S_k(y) = \min_{0 \leq x \leq \min\{y, v_k\}} [S_{k-1}(y-x) + h_k(x)], k=2, \dots, 5; y=0, 1, \dots, 8.$$

Прямой ход. В результате прямого хода заполняем таблицу 2.4, в которую помещены значения  $S_k(y)$ , а через дробь (/) указаны условно-оптимальные решения, которые помогут восстановить оптимальное решение на этапе обратного хода.

$y$	$S_1(y)$	$S_2(y)$	$S_3(y)$	$S_4(y)$	$S_5(y)$
0	<b>0</b>	0	0	0	
1	2.5/1	2.5/0	2.2/1	2/1	
2	5/2	4/2	4/0	4/0	
3	-	<b>6/3</b>	<b>6/0</b>	6/0	
4	-	8.5/3	8.2/1	8/1	
5	-	11/3	10.4/2	10/2	
6	-	-	12.6/3	12/3	
7	-	-	15.1/3	14/4	
8	-	-	17.6/3	<b>16/5</b>	<b>16/0</b>

Таб. 2.4.

Здесь следует сделать несколько замечаний:

1. Прочерк в клетке таблицы следует считать «бесконечностью», т.к. при соответствующих значениях параметров допустимого решения не существует.
2. Условно-оптимальных решений (как и оптимальных) может быть несколько. Для получения *одного* из оптимальных решений достаточно хранить *любое* условно-оптимальное решение.
3. Значения функции  $S_5(y)$  для  $y=0, 1, \dots, 7$  вычислять нет необходимости. Эти значения понадобились бы при вычислении  $S_6(y)$ , но в этом нет необходимости т.к. всего 5 складов.

Итак, в результате работы прямого хода алгоритма найдено оптимальное значение целевой функции  $S^*=S_5(8)=16$  и оптимальное значение последней переменной  $x_5^*=x_5(8)=0$ .

Обратный ход. Так как на пятый склад в оптимальном решении не надо везти ни одного контейнера ( $x_5^*=0$ ), то их нужно развести по первым четырем складам. Следовательно,  $y_4^*=8-x_5^*=8$ . Значит, для определения оптимального значения предпоследней переменной достаточно обратиться к клетке таблицы 2.4 со значением  $S_4(8)$ . В этой клетке хранится условно-оптимальное значение  $x_4(8)$ . Имеем  $x_4^*=x_4(8)=5$ . Значит, на склад 4 будет отправлено 5 контейнеров. Следовательно, еще 3 нужно развести по первым трем складам, т.е.  $y_3^*=8-x_4^*=3$ . Клетка  $S_3(3)$  таблицы 2.4 хранит условно-оптимальное значение  $x_3(3)=0$ . Поэтому  $x_3^*=0$  и  $y_2^*=3-x_3^*=3$ . В клетке таблицы 2.4 со значением  $S_2(y_2^*)=S_2(3)$  хранится также значение условно оптимальной переменной  $x_2(3)=3$ . Значит,  $x_2^*=3$  и  $y_1^*=3-x_2^*=0$ , т.е. все контейнеры распределены и на склад 1 ни один из них не повезут.

Окончательно имеем, что оптимальный вектор рассматриваемой задачи  $x^*=(0,3,0,5,0)$ , приводит к минимальным затратам  $S^*=16$ , связанным с перевозкой и хранением 8 контейнеров. В таблице 2.4 помечены ячейки, по значениям которых было восстановлено оптимальное решение на этапе обратного хода алгоритма.

Заметим, что для вычисления значений  $S_k(y)$  нужны значения лишь предыдущего столбца  $S_{k-1}(y)$ . Не обязательно хранить всю таблицу. Правда, при этом мы потеряем значения условно-оптимальных решений, что приведет к увеличению трудоемкости. Таким образом, можно реализовать вариант алгоритма с трудоемкостью  $O(n^2Y^2)$ , уменьшив требуемую память до величины  $O(Y)$ . Такая реализация называется *релаксационным алгоритмом* ДП. Она состоит из  $(n-1)$ -го прямого хода решения задач  $\langle k, f, y_k^* \rangle$ ,  $k=n, n-1, \dots, 2$ . В результате первого прямого хода будет найдено оптимальное значение целевой функции исходной задачи  $\langle n, f, Y \rangle$  и последняя компонента вектора решения  $x_n^*$ , что позволит определить  $y_{n-1}^* = Y - x_n^*$ . Затем решаем задачу  $\langle n-1, f, y_{n-1}^* \rangle$  и получаем  $x_{n-1}^*$ . Продолжая выполнение прямого хода для задач  $\langle k, f, y_k^* \rangle$ ,  $k=n-2, \dots, 2$ , определим оптимальное решение исходной задачи.

Пример 2.3. Решить задачу из примера 2.2 релаксационным алгоритмом ДП.

Решение. Выполняя прямой ход (без хранения таблицы 2.4) для исходной задачи  $\langle 5, h, 8 \rangle$ , найдем  $S^* = 16$  и  $x_5^* = 0$ . Значит,  $y_4^* = 8 - 0 = 8$ . Теперь выполним прямой ход для задачи  $\langle 4, h, 8 \rangle$ . Получим  $x_4^* = 5$ . Следовательно,  $y_3^* = 8 - 5 = 3$ . Выполним прямой ход для задачи  $\langle 3, h, 3 \rangle$ . Имеем  $x_3^* = 0$ . Значит,  $y_2^* = 3 - 0 = 3$ . Выполним прямой ход для задачи  $\langle 2, h, 2 \rangle$  и получим  $x_2^* = 3$ . Очевидно,  $y_1^* = 3 - 3 = 0$  и  $x_1^* = 0$ .

## 2.2. Задача о ранце

Задача о ранце (ЗР) формулируется следующим образом. Пусть имеется множество типов предметов  $k=1, \dots, n$ , каждый из которых имеет объем (вес)  $a_k \geq 0$  и ценность  $f_k(x_k)$ . Требуется заполнить ранец предметами, суммарная ценность которых максимальна, а суммарный объем не превосходит емкости ранца  $A$ . Задачу, математическая модель которой имеет вид

$$\sum_{k=1}^n f_k(x_k) \rightarrow \max_{x=(x_1, \dots, x_n) \in Z_+^n};$$

$$\sum_{k=1}^n a_k x_k \leq A,$$

обозначим  $\langle n, f, \leq A \rangle$  и поместим в семейство подобных задач  $\{\langle k, f, \leq \alpha \rangle, k=1, \dots, n, 0 \leq \alpha \leq A\}$ . Пусть  $S_k(\alpha)$  – оптимальное значение целевой функции задачи  $\langle k, f, \leq \alpha \rangle$ .

Справедливы рекуррентные соотношения

$$S_1(\alpha) = \max_{x_1=0, \dots, [\alpha/a_1]} f_1(x_1), \quad 0 \leq \alpha \leq A;$$

$$S_k(\alpha) = \max_{x_k=0, \dots, [\alpha/a_k]} \{S_{k-1}(\alpha - a_k x_k) + f_k(x_k)\}, \quad k=2, \dots, n, \quad 0 \leq \alpha \leq A,$$

которые верны для любых  $a_k$  и  $f_k$ . Для численной реализации схемы достаточно предположить целочисленность  $a_k$ . В этом случае  $\alpha \in \{0, \dots, A\}$ , где  $A$  также можно считать целым числом.

Подробнее рассмотрим линейную задачу о ранце, которая имеет вид:

$$(I) \begin{cases} \sum_{k=1}^n c_k x_k \rightarrow \max_{\{x_k\}}; \\ \sum_{k=1}^n a_k x_k \leq A; \\ x_k \in Z_+, k = 1, \dots, n. \end{cases}$$

Обозначим эту задачу  $\langle n, \leq A \rangle$  и поместим в семейство  $\{\langle n, \leq \alpha \rangle, 0 \leq \alpha \leq A\}$ . Пусть  $S(\alpha)$  – оптимальное значение функционала задачи  $\langle n, \leq \alpha \rangle$ .

Теорема 2.2. Справедливо соотношение

$$S(\alpha) = \max_{k=1, \dots, n | a_k \leq \alpha} \{S(\alpha - a_k) + c_k\}, \quad 0 \leq \alpha \leq A. \quad (2.6)$$

Когда в результате прямого хода значения  $S(\alpha)$ ,  $\alpha=0, \dots, A$  вычислены, можно восстановить оптимальный вектор  $x^*$  следующим образом. Сначала положим  $x^*=0$ ,  $\alpha^*=A$ . Затем последовательно выполним шаги, на каждом из которых найдем индекс  $k$ , при котором выполняется равенство  $S(\alpha^*) = S(\alpha^* - a_k) + c_k$ ,  $a_k \leq \alpha^*$ . Положим  $x_k^* = x_k^* + 1$ ;  $\alpha^* = \alpha^* - a_k$  и повторим шаг. Если допустимого индекса  $k$  нет, тогда полученный вектор  $x^*$  оптимален.

Использование рекуррентных соотношений (2.6) в случае целочисленных  $a_k$ , приводит к трудоемкости  $O(An)$  и объему требуемой памяти  $O(A+n)$ .

Наряду с задачей (I) рассмотрим задачу

$$(II) \begin{cases} \sum_{k=1}^n a_k x_k \rightarrow \min_{\{x_k\}}; \\ \sum_{k=1}^n c_k x_k \geq B; \\ x \in Z_+^n, \end{cases}$$

которую обозначим  $\langle n, \geq B \rangle$  и назовем *обратной* к задаче (I). Так же как и ранее, поместим задачу (II) в семейство  $\{\langle n, \geq \beta \rangle, 0 \leq \beta \leq B\}$  и обозначим через  $Q(\beta)$  оптимальное значение целевой функции, а через  $x^0(\beta)$  – оптимальное решение задачи  $\langle n, \geq \beta \rangle$ . Справедливы рекуррентные соотношения:

$$Q(0) = 0;$$



$$Q(\beta) = \min_{1 \leq k \leq n} \{Q(\max\{0, \beta - c_k\}) + a_k\}, 0 \leq \beta \leq B.$$

Лемма 2.1. Функция  $Q(\beta)$  не убывает.

Теорема 2.3. Пусть  $\tilde{\beta} = \max\{\beta \mid Q(\beta) \leq A, \beta \geq 0\}$ . Тогда  $S(A) = \tilde{\beta}$  и оптимальное решение  $x^0(\tilde{\beta})$  задачи  $\langle n, \geq \tilde{\beta} \rangle$  является также оптимальным решением задачи  $\langle n, \leq A \rangle$ .

Из неубывания функции  $Q(\beta)$  и утверждений теоремы следует, что

$$S^* = \min\{\beta \mid A < Q(\beta+1), \beta = 0, 1, \dots\}.$$

Это позволяет построить другой алгоритм решения задачи (I):

- на этапе прямого хода находим элементы таблицы  $Q(\beta)$  по рекуррентным соотношениям для задачи (II), последовательно полагая  $\beta = 0, 1, \dots$ , пока для некоторого  $\tilde{\beta}$  не получим  $A < Q(\tilde{\beta} + 1)$ . Это значение  $\tilde{\beta}$  и есть оптимальное значение целевой функции  $S^*$  задачи  $\langle n, \leq A \rangle$ ;
- по полученной таблице  $\{Q(\beta), \beta=0, 1, \dots, S^*\}$  находим оптимальное решение  $x^0(S^*)$  обратной задачи  $\langle n, \geq S^* \rangle$ , совпадающее с искомым решением  $x^*(A)$  прямой задачи  $\langle n, \leq A \rangle$ .

Справедлива также следующая

Теорема 2.4. Пусть  $\tilde{\alpha} = \min\{\alpha \mid S(\alpha) \geq B, \alpha \geq 0\}$ . Тогда  $Q(B) = \tilde{\alpha}$  и оптимальное решение  $x^*(\tilde{\alpha})$  прямой задачи  $\langle n, \leq \tilde{\alpha} \rangle$  является также оптимальным решением  $x^0(B)$  обратной задачи  $\langle n, \geq B \rangle$ .

Утверждения последних двух теорем позволяют осуществлять переход от прямой задачи к обратной и наоборот. В каких случаях этот переход оправдан? Отметим два случая:

1. Если в прямой (обратной) задаче параметры  $a_k$  ( $c_k$ ) не целочисленные, а  $c_k$  ( $a_k$ ) целочисленные;
2. Если в прямой (обратной) задаче число  $A$  ( $B$ ) большое, и  $a_k$  ( $c_k$ ) достаточно большие параметры, чтобы быстро выполнялось неравенство  $A < Q(\beta+1)$  ( $S(\alpha-1) < B$ ).

В первом случае переход к обратной (прямой) задаче необходим для конечности реализации алгоритма. В случае 2 переход к обратной (прямой) задаче может уменьшить трудоемкость.

Проиллюстрируем решение прямой задачи (ПЗ) путем перехода к обратной задаче (ОЗ) на примере булевой задачи о ранце. Запишем обе задачи:

$$\begin{aligned}
 (\text{ПЗ}) \quad & \begin{cases} \sum_{i=1}^n r_i x_i \rightarrow \max_{\{x_i\}}; \\ \sum_{i=1}^n h_i x_i \leq b; \\ x_i \in \{0,1\}, i = 1, \dots, n. \end{cases} & (\text{ОЗ}) \quad & \begin{cases} \sum_{i=1}^n h_i x_i \rightarrow \min_{\{x_i\}}; \\ \sum_{i=1}^n r_i x_i \geq d; \\ x_i \in \{0,1\}, i = 1, \dots, n. \end{cases}
 \end{aligned}$$

Так как переменные принимают два значения, рекуррентные соотношения для ОЗ запишем в виде:

$$\begin{aligned}
 Q_i(y) &= \begin{cases} 0, & y = 0; \\ h_i, & 0 < y \leq r_i; \\ +\infty, & y > r_i; \end{cases} \\
 Q_k(y) &= \min \left\{ \begin{aligned} & Q_{k-1}(y), \quad x_k = 0; \\ & h_k + Q_{k-1}(\max\{0, y - r_k\}), \quad x_k = 1; \end{aligned} \right\}, \quad k=2, \dots, n, y=0, \dots, d.
 \end{aligned}$$

Величины  $Q_k(y)$ , согласно приведенным выше утверждениям, нужно вычислять, увеличивая  $y=0,1,\dots$ , пока не будет выполнено неравенство  $A < Q(\beta+1)$ . Тогда оптимальное значение функционала ПЗ  $S^*=\beta$  и решение ОЗ с  $d=\beta$  будет оптимальным и для ПЗ. Разберем следующий

Пример 2.4. Пусть параметры ПЗ заданы таблицей 2.5, а  $b=110$ .

$i$	1	2	3	4	5	6
$r_i$	6	5	4	3	2	1
$h_i$	45	33	28	16	13	9

Таб. 2.5.

Решение. Решение ПЗ приведет к заполнению таблицы размерности  $6 \times 110$ . Чтобы избежать громоздких расчетов, перейдем к ОЗ и в процессе работы прямого хода алгоритма ДП вычислим  $Q_k(y)$ ,  $k=1,\dots,6$  для тех  $y=0,1,\dots$ , для которых выполняются неравенства  $Q_k(y) \leq 110$ . Прочерками в таблице обозначим бесконечно большие величины (когда нет допустимых решений). В столбцах, соответствующих  $Q_4, Q_5, Q_6$ , вычисления прекращены, как только соответствующие величины стали больше  $b=110$ . Кроме значений  $Q_k(y)$ , будем хранить в таблице 2.6 также (через дробь) в соответствующей ячейке значение условно-оптимального решения. Если

условно-оптимальных решений несколько, то мы здесь запоминаем только одно из них.

$y$	$Q_1$	$Q_2$	$Q_3$	$Q_4$	$Q_5$	$Q_6$
0	0	0	0	0	0	0
1	45/1	33/1	28/1	16/1	13/1	9/1
2	45/1	33/1	28/1	16/1	13/1	13/0
3	45/1	33/1	28/1	16/1	16/0	16/0
4	45/1	33/1	28/1	28/1	28/0	25/1
5	45/1	33/1	33/1	33/1	29/1	29/0
6	45/1	45/1	45/1	44/1	41/1	38/1
7	-	78/1	61/1	44/1	44/0	44/0
8	-	78/1	61/1	49/1	49/0	49/0
9	-	78/1	61/1	61/0	57/1	57/0
10	-	78/1	73/1	73/0	62/1	62/0
11	-	78/1	78/0	77/1	74/1	71/1
12	-	-	106/1	77/1	77/0	77/0
13	-	-	106/1	89/1	89/0	86/1
14	-	-	106/1	94/1	90/1	90/0
15	-	-	106/1	106/0	101/1	99/1
16	-	-	-	122/1	107/1	107/0
17	-	-	-	-	119/1	116/0

Таб. 2.6.

Обратный ход. Максимальное значение  $y$ , при котором  $Q_6(y)=107 \leq b=110$ , равно 16. Следовательно, оптимальное значение целевой функции ПЗ равно  $S^*=16$ . При этом условно-оптимальное решение  $x_6^0(16)=0$  определяет значение последней компоненты оптимального вектора ПЗ  $x_6^*=0$ . Отсюда следует, что  $\sum_{i=1}^5 r_i x_i \geq 16$ . Следовательно, для определения  $x_5^*$  следует воспользоваться ячейкой  $Q_5(16)$ , в которой условно-оптимальное решение определяет  $x_5^*=1$ . Значит,  $\sum_{i=1}^4 r_i x_i \geq 16 - r_5 = 16 - 2 = 14$ . Переходим в ячейку  $Q_4(14)$  таблицы 2.6, откуда определяем  $x_4^*=1$ . Следовательно,  $\sum_{i=1}^3 r_i x_i \geq 14 - r_4 = 14 - 3 = 11$ . Значение  $x_3^*=0$  оставляет нас в той же строке таблицы, но переводит в предыдущий столбец, из которого

имеем  $x_2^* = 1$ . Значит,  $r_1 x_1 \geq 11 - r_2 = 11 - 5 = 6$ . Шестая строка первого столбца определяет первую компоненту оптимального вектора  $x_1^* = 1$  и решение задачи  $x^* = (1, 1, 0, 1, 1, 0)$ .

### 2.3. Задача о ближайшем соседе

Рассмотрим проблему оптимального разбиения линейного объекта на участки, известную под названием *задача о ближайшем соседе*. Пусть задано целое положительное число  $M$  и неотрицательная функция  $f(x, y)$ , которая отражает затраты, связанные с «обслуживанием» отрезка  $[x, y] \subseteq [0, M]$ . Требуется разбить отрезок  $[0, M]$  на  $n$  частей таким образом, чтобы суммарные затраты, соответствующие этому разбиению были минимальны. Ниже приведена математическая постановка задачи

$$\begin{cases} \sum_{k=1}^n f(x_{k-1}, x_k) \rightarrow \min; \\ 0 = x_0 \leq x_1 \leq \dots \leq x_n = M, \end{cases}^x$$

которую обозначим  $\langle n, M \rangle$ .

Пусть  $S_n(M)$  – оптимальное значение целевой функции задачи  $\langle n, M \rangle$ , которую поместим в семейство задач  $\{\langle k, y \rangle, k=1, \dots, n, y=1, \dots, M\}$ . Пусть  $S_k(y)$  – оптимальное значение целевой функции задачи  $\langle k, y \rangle$ .

Справедливы рекуррентные соотношения:

$$S_k(y) = \begin{cases} f(0, y), & k=1, \quad y=1, \dots, M; \\ \min_{0 \leq x \leq y} \{S_{k-1}(x) + f(x, y)\}, & k=2, \dots, n; \quad y=1, \dots, M. \end{cases}$$

**Пример 2.5.** Решить задачу  $\langle 4, 8 \rangle$  с функций  $f(x, y)$ , заданной таблицей 2.7.

$x \backslash y$	1	2	3	4	5	6	7	8
0	3	19	24	41	42	63	66	83
1	0	6	18	26	39	48	56	77
2	-	0	11	19	35	44	55	56
3	-	-	0	13	25	27	45	53
4	-	-	-	0	3	15	24	37
5	-	-	-	-	0	3	16	27
6	-	-	-	-	-	0	12	21
7	-	-	-	-	-	-	0	16
8	-	-	-	-	-	-	-	0

Таб. 2.7.

Решение. Прямой ход. Пользуясь рекуррентными соотношениями, заполним таблицу 2.8, в каждой ячейке которой поместим соответствующее значение  $S_k(y)$ ,  $k=1, \dots, 4$ ,  $y=0, \dots, 8$  и (через дробь /) условно-оптимальное решение  $x(y)$ , которое соответствует самой правой точке оптимального разбиения отрезка  $[0, y]$ .

$y$	$S_1(y)$	$S_2(y)$	$S_3(y)$	$S_4(y)$
0	0	0	0	0
1	<b>3</b>	3/0	3/0	3/0
2	19	9/1	9/1	9/1
3	24	21/1	20/2	20/2
4	41	<b>29/1</b>	28/2	28/2
5	42	42/0	<b>32/4</b>	31/4
6	63	45/5	44/4	35/5
7	66	58/5	53/4	48/5
8	83	69/5	65/2	<b>59/5</b>

Таб. 2.8.

Обратный ход. На последнем шаге прямого хода найдены минимальные затраты  $S^*=S_4(8)=59$ , соответствующие оптимальному разбиению исходного отрезка  $[0, 8]$ , и оптимальное значение последней точки разбиения  $x_3^*=5$ . Следовательно, теперь отрезок  $[0, 5]$  необходимо оптимально разбить на *три* части. Значение  $x_2^*$  получим как условно-оптимальное решение, соответствующее величине  $S_3(5)$  из таблицы 2.8. Значит,  $x_2^*=4$ . Далее, последней точкой оптимального разбиения отрезка  $[0, 4]$  является  $x_1^*=1$ . В результате получили следующие точки оптимального разбиения исходного отрезка  $[0, 8]$  на 4 части:  $x_0^*=0$ ,  $x_1^*=1$ ,  $x_2^*=4$ ,  $x_3^*=5$ ,  $x_4^*=8$ . В таблице 2.8 выделены ячейки, по значениям которых восстанавливается оптимальное решение.

Методом ДП можно решить также задачу о ближайшем соседе, в которой число отрезков разбиения  $n$  не задано:

$$\begin{cases} \sum_{k=1}^n f(x_{k-1}, x_k) \rightarrow \min_{x, n} \\ 0 = x_0 < x_1 < \dots < x_n = M, \quad n > 0, \end{cases}$$

которую обозначим  $\langle M \rangle \in \{ \langle y \rangle | y=1, \dots, M \}$ . Эту задачу можно свести к последовательности задач  $\langle n, M \rangle$ ,  $n=1, \dots, M$ , вычислить  $S_n(M)$  и найти  $n^* = \arg \min \{ S_n(M), n=1, \dots, M \}$ . Такой подход имеет вычислительную сложность

$O(M^3)$  и требует память –  $O(n^*M)$ . Однако для нее справедливы простые рекуррентные соотношения:

$$\begin{aligned}\tilde{S}(0) &= 0; \\ \tilde{S}(y) &= \min_{x=0,1,\dots,y-1} \{ \tilde{S}(x) + f(x,y) \}, y = 1, \dots, M.\end{aligned}$$

Заметим, что в постановке задачи мы заменили нестрогие неравенства на строгие. Это можно сделать, если  $f(x,x) \geq 0$ . Таким образом, в случае, когда оптимальное решение не единственно, будет найдено то из них, в котором  $n$  минимально.

Пример 2.6. Решить задачу из примера 2.5 с произвольным  $n > 0$ .

Решение. Прямой ход. Вычислим значения  $\tilde{S}(y)$  для  $y=0,1,\dots,8$  и напомним, как и прежде, соответствующие условно-оптимальные решения (см. таблицу 2.9).

$y$	0	1	2	3	4	5	6	7	8
$\tilde{S}(y)$	0	3/0	9/1	20/2	28/2	31/4	34/5	46/6	55/6

Таб. 2.9.

Обратный ход. Имеем минимальные затраты, связанные с разбиением отрезка  $[0,8]$  на оптимальное количество частей,  $S^* = \tilde{S}(M) = \tilde{S}(8) = 55$  и последнюю точку оптимального разбиения, равную 6. Здесь необходимо отметить, что количество точек оптимального разбиения неизвестно, и мы пока не можем сказать, какой по счету является точка 6. Следующей (справа) точкой разбиения является 5 (см. столбец 6 таблицы 2.9), затем – 4, 2 и 1. Значит,  $x_0^* = 0$ ,  $x_1^* = 1$ ,  $x_2^* = 2$ ,  $x_3^* = 4$ ,  $x_4^* = 5$ ,  $x_5^* = 6$ ,  $x_6^* = 8$ . Следовательно,  $n^* = 6$ .

Если  $n$  является переменной, для которой дополнительно требуется выполнение неравенств  $a \leq n \leq b$ , то такую задачу о ближайшем соседе следует решать в четыре этапа:

Этап 1. Решить задачу с произвольным  $n > 0$ . Получим некоторое оптимальное значение  $n^*$ . Если  $n^* \in [a, b]$ , то задача решена. Иначе переходим к выполнению следующего этапа.

Этап 2. Выполнить прямой ход для задачи с фиксированным значением  $n = b$ .

Этап 3. Найти  $n^*$ , для которого  $S_{n^*}^*(M) = \min_{a \leq n \leq b} S_n(M)$ .

Этап 4. Для восстановления оптимального решения осуществить обратный ход при  $n = n^*$ .

Пример 2.7. Решить задачу из примера 2.5 с количеством отрезков разбиения  $n$ , удовлетворяющим неравенствам  $2 \leq n \leq 5$ .

Решение. Этап 1. Из предыдущего примера имеем, что  $n^*=6$ , поэтому  $n^* \notin [2, 5]$ . Переходим к следующему этапу.

Этап 2. Положим  $n=b=5$  и выполним прямой ход для задачи  $\langle 5, 8 \rangle$ . Результат помещен в таблице 2.10, в которой, как и прежде, кроме значений  $S_k(y)$ ,  $k=1, \dots, 5$ ,  $y=0, \dots, 8$ , в соответствующие ячейки (через дробь /) помещены условно-оптимальные решения.

$y$	$S_1(y)$	$S_2(y)$	$S_3(y)$	$S_4(y)$	$S_5(y)$
0	0	0	0	0	
1	3	3/0	3/0	3/0	
2	19	9/1	9/1	9/1	
3	24	21/1	20/2	20/2	
4	41	29/1	28/2	28/2	
5	42	42/0	32/4	31/4	
6	63	45/5	44/4	35/5	
7	66	58/5	53/4	48/5	
8	83	69/5	65/2	59/5	56/6

Таб. 2.10.

Этап 3. Найдем  $n^*$ , для которого  $S_{n^*}(8) = \min_{n=2, \dots, 5} S_n(8) = \min\{69, 65, 59, 56\} = 56$ . Имеем  $n^*=5$ .

Этап 4. Обратный ход для  $n^*=5$  позволяет восстановить оптимальное разбиение для последней задачи:  $x_0^*=0$ ,  $x_1^*=1$ ,  $x_2^*=4$ ,  $x_3^*=5$ ,  $x_4^*=6$ ,  $x_5^*=8$ . В таблице 2.10 выделены ячейки, значения которых позволяют осуществить обратный ход и получить оптимальное решение.

### Упражнения.

1. Решить распределительную задачу, в которой  $f_i(x_i) = f(x_i)$ ,  $i=1, \dots, n$  и функция  $f$  – строго выпуклая и дифференцируемая.
2. Решить линейную задачу о ранце без требования целочисленности переменных.
3. Имеется восемь предметов, каждый из которых характеризуется своей ценностью и весом. Определить набор предметов максимальной ценности, общий вес которых не превосходит 55, если исходные данные заданы следующей таблицей.

Номер предмета	1	2	3	4	5	6	7	8
вес предмета	2	8	17	4	26	2	23	9
стоимость предмета	4	2	3	1	5	2	10	8

4. Завод может производить 4 вида продукции. Обозначим  $g_i(x)$  – количество единиц сырья для производства  $x$  единиц продукции  $i$ -го вида,  $c_i$  – доход от реализации единицы продукции  $i$ -го вида. Пусть всего имеется 50 единиц сырья. Требуется определить, сколько производить продукции каждого типа, чтобы максимизировать доход, если доходы от реализации единицы продукции заданы таблицей

$i$	1	2	3	4
$c_i$	2	3	1	1

а расход сырья на единицу продукции – таблицей

$x$	$g_1$	$g_2$	$g_3$	$g_4$
1	15	20	20	15
2	25	30	21	18
3	35	35	22	20
4	44	40	23	23
5	48	50	28	25

5. Разбить отрезок  $[0,8]$  с минимальными затратами на не более чем четыре части, если затраты  $f(x,y)$ , связанные с отрезком  $[x,y]$  характеризуются следующей таблицей

$x \backslash y$	0	1	2	3	4	5	6	7	8
0	0	0	14	21	95	77	58	81	83
1	-	0	7	8	26	100	59	60	98
2	-	-	0	9	5	1	88	98	64
3	-	-	-	0	0	8	1	78	97
4	-	-	-	-	0	1	9	10	82
5	-	-	-	-	-	0	2	0	0
6	-	-	-	-	-	-	0	2	8
7	-	-	-	-	-	-	-	0	8
8	-	-	-	-	-	-	-	-	0

6. На рисунке изображено дерево  $T=(V,E)$  с корнем в  $r \in V$  и весами вершин  $c_v$ ,  $v \in V$ . С помощью динамического программирования найти поддерево с корнем в  $r$  максимального веса.



