2.3. Дискретное программирование

В общем случае задача дискретного программирования имеет следующую постановку [2, 3, 10, 41, 63].

Найти максимум (минимум) целевой функции $f(x_1, x_2, ..., x_n)$ при заданных условиях

$$\left.\begin{aligned} & g_1(x_1, x_2, ..., x_n) \le b_1, \\ & g_2(x_1, x_2, ..., x_n) \le b_2, \\ & ... \\ & g_m(x_1, x_2, ..., x_n) \le b_m, \\ & X = \{x_1, x_2, ..., x_n\} \subseteq D, \end{aligned}\right\} \qquad (2.22)$$

где D – конечное или счётное множество.

В этом случае говорят о ***дискретном программировании***. Если X ограничено множеством целых чисел, то задачу назначают задачей ***линейного целочисленного программирования (ЛЦП)***.

Известны следующие классы задач дискретного программирования.

- Задачи с неделимостями (задачи о рюкзаке), обусловлены физическими свойствами объектов. Это задача о размещении массивов информации на внешних устройствах ЭВМ при ограничениях на объём, скорость вращения, стоимостные рамки и др.
- Экстремальные комбинаторные задачи (выбора, о назначениях, коммивояжёра, о покрытиях).
- Задачи на несвязных и невыпуклых областях.
- Задачи с разрывной целевой функцией.
- Транспортная задача. Когда задано ограничение целочисленности, то есть при целых значениях массивов поставок, потребления и стоимостей или ограничениях на пропускные способности коммуникаций, является задачей линейного целочисленного программирования.

Для решения задач дискретного программирования применяются как строго обоснованные, так и неформальные метода поиска решений.

1. Метод отсечений, отсекающих плоскостей, он же метод ГОмори (или ГомОри).
2. Метод ветвей и границ.
3. Методы, учитывающие особенности задачи.
4. Методы случайного поиска (эвристические).

2.3.1. Решение задачи ЛЦП методом Гомори

Данный метод носит название метода ***отсекающих плоскостей*** или метода ***целочисленных форм***, но чаще именуется по имени автора. В основании метода положены следующие теоретические положения [8, 10].

Любое уравнение или неравенство линейной системы ограничений представимо линейной комбинацией базисных векторов и в канонической форме записывается так:

$$\sum_{j=1}^{n} d_{ij} \cdot x_j = p_i,\ i = \overline{1,m}. \tag{2.23}$$

Обозначим заключением в квадратные скобки $[d]$ целую часть d:

$[2{,}5] = 2$; $[10/3] = 3$; $[4] = 4$; $[-2{,}5] = -3$; $[-10/3] = -4$; $[-4] = -4$.

По сути дела, операция […] представляет собой округление по недостатку (в меньшую сторону).

Если считать переменные $X = (x_1,\ x_2,\ \ldots\ ,\ x_n)$ целыми числами, то можно перейти к более слабому, по сравнению с (2.23), выражению

$$\sum_{j=1}^{n} \left[d_{ij}\right] \cdot x_j \le p_i,\ i = \overline{1,m}, \tag{2.24}$$

а учитывая, что сумма в (2.20) целочисленна, то справедливо и

$$\sum_{j=1}^{n} \left[d_{i,j}\right] \cdot x_j \le [p_i],\ i = \overline{1,m}. \tag{2.25}$$

Введя свободную целочисленную переменную x_{n+t}, канонизируем (2.25):

$$\sum_{j=1}^{n} \left[d_{ij}\right] \cdot x_j + x_{n+t} = [p],\ i = \overline{1,m}. \tag{2.26}$$

Очевидно, что добавление последнего равенства к исходной канонизированной системе ***не противоречит*** исходной системе ограничений.

Чтобы получить (2.26) из (2.23), необходимо «отсечь» от (2.26) дробную часть. С этой целью формируется ***отсечение*** для приведения произвольного уравнения в целочисленную форму. Указанное отсечение представляет собою уравнение, которое, будучи прибавленным к исходному уравнению, делает его целочисленным:

$$\sum_{j=1}^{n}\{-d_{ij}\}\cdot x_j + x_{n+t} = \{-p\},\ i=\overline{1,m}. \qquad (2.27)$$

В (2.27) обозначена символами {…} операция нахождения дробной части. Отсюда просматривается простой вычислительный алгоритм.

Алгоритм формирования отсечения Гомори

1. Для выбранного канонизированного уравнения (2.23) сформировать желаемую целочисленную форму вида (2.26).

2. Из целочисленной формы (2.26) вычесть исходное уравнение (2.23), получится уравнение отсекающей плоскости (2.27).

Дадим и нотацию этого алгоритма в виде формулы:

<Отсечение> = <Целочисленная_форма> – <Исходная_строка>. (2.28)

Алгоритм решения задачи ЛЦП методом Гомори

Процедура получения решения структурно состоит из

- предварительного этапа,
- проверки условия окончания и
- так называемой “большой итерации”, которая включает операцию формирования отсечения и несколько шагов итеративной части двойственного симплекс-метода.

1. ***Предварительный этап***. В ходе его получают оптимальное решение ЗЛП без учёта целочисленности. Решение выполняется любым удобным методом, кроме, разумеется, графического метода.

2. ***Условие окончания*** расчётов. Если в текущем решении в***се компоненты базисного столбца*** A_0, соответствующие основным переменным, ***являются целыми*** числами, то найдено оптимальное решение задачи ЛЦП. В противном случае, выполняется большая итерация.

3. ***Большая итерация***.

3.1. ***Отсечение Гомори*** формируется

- ***для тех строк*** симплекс-таблицы, ***в которых*** компоненты $a_{i,0}$, соответствующие основным переменным задачи, ***дробные числа***;
- ***на*** каждом ***шаге*** алгоритма отсечение Гомори формируется только ***по одной*** из строк;
- ***очерёдность*** формирования отсечений ***не регламентируется***.

3.2. В базисное решение вводится дополнительная переменная x_{r+t}, соответствующая канонизированному уравнению отсекающей

плоскости, одновременно симплекс-таблица пополняется строкой и столбцом-ортом A_{n+t}.

3.3. Выполняется итерационная часть двойственного симплекс-метода. Направляющая строка соответствует вектору A_{n+t}, а направляющий столбец определяется по обычному условию этого метода:

$$\arg\min_{j}\left\{\frac{-\delta_j \geq 0}{a_{i^*,j} < 0}\right\} \Rightarrow j^*.$$

3.4. Если вектор A_{n+t}, ранее выведенный из базиса, в ходе расчётом снова в него вводится в процессе итераций, то строку и столбец симплекс-таблицы, соответствующие переменной x_{n+t} после пересчёта по методу Жордана-Гаусса вычёркивают (удаляют) из неё.

На этом циклическая часть алгоритма завершена, а цикл возобновляется с п. 2.

Замечания к методу Гомори [10].

1. Сходимость вычислений обеспечивается за конечное число итераций, что и обусловливает применение данного метода на практике

2. Метод особенно эффективен, когда ***большинство*** переменных в оптимальном нецелочисленном решении имеют целочисленные значения.

4. После выполнения нескольких больших итераций на шаге отсечения Гомори появляются многочисленные альтернативы. Это ведёт к зацикливанию, именуемому Г. Вагнером [10] “сплошной вырожденностью”, когда решение возвращается к ранее бытовавшей позиции, вследствие неверного выбора строки для формирования отсечения. Вагнеру известны многочисленные примеры, когда при значениях n и m, не превышающих десяти, потребовались тысячи итераций, прежде чем оптимум был достигнут.

5. Затруднена сходимость при решении задач, в которых значения элементов $a_{i,j}$ и b_i велики.

6. Иногда, для достижения успеха, требуется видоизменить постановку задачи в сторону усиления, например, введя ограничения $x_1 \leq 6$ и $x_2 \leq 6$ в дополнение к уже существующему ограничению $x_1 + x_2 \leq 6$.

Рассмотрим пример применения метода Гомори.

Поскольку операции с симплекс-таблицами нами ранее вельми подробно рассматривались, ограничим содержание нашего примера процессом формирования отсечений Гомори по результатам решения демонстрационной задачи о производстве изделий из картошки.

Оптимальное решение этой задачи без учёта ограничения целочисленности имеет вид:

		c_j	5	6	0	0	0
Базис	$C_Б$	A_0	A_1	A_2	A_3	A_4	A_5
A_2	6	3	0	1	5	–5	0
A_1	5	4,5	1	0	–2,5	7,5	0
A_5	0	0,15	0	0	–0,75	–0,75	1
	δ_j	40,5	0	0	17,5	7,5	0

Видно, что основная переменная x_1 нецелая, поэтому необходимы отсечения.

Сформируем отсечение Гомори по второй строке, которая соответствует основной переменной x_1. Исходной строке соответствует уравнение

$$4{,}5 = 1x_1 + 0x_2 - 2{,}5x_3 + 7{,}5x_4 + 0x_5 .$$

Целочисленная форма для этой строки есть

$$4 = 1x_1 + 0x_2 - 3x_3 + 7x_4 + 0x_5 .$$

Поэтому отсечение будет

$$\begin{array}{r} 4 = 1x_1 + 0x_2 - 3x_3 + 7x_4 + 0x_5 \\ - \\ 4{,}5 = 1x_1 + 0x_2 - 2{,}5x_3 + 7{,}5x_4 + 0x_5 \\ \hline -0{,}5 = 0x_1 + 0x_2 - 0{,}5x_3 - 0{,}5x_4 + 0x_5 \end{array} .$$

Результат вычислений занесём в симплекс-таблицу в отдельную строку. Одновременно таблица пополнится дополнительным столбцом для вектора A_6 соответствующим переменной x_6.

			c_j	5	6	0	0	0	0
	Базис	$C_Б$	A_0	A_1	A_2	A_3	A_4	A_5	A_6
	A_2	6	3	0	1	5	–5	0	0
	A_1	5	4,5	1	0	–2,5	7,5	0	0
	A_5	0	0,15	0	0	–0,75	–0,75	1	0
←	A_6	0	–0,5	0	0	–0,5	–0,5	0	1
		δ_j	40,5	0	0	17,5	7,5	0	0
							↑		

Далее выполняются несколько итерационных шагов двойственного симплекс-метода:

- выводимая строка определяется отрицательной компонентой столбца A_0;
- вектор, вводимый в базис определяет условие $\min\left\{\frac{-17,5}{-0,5};\frac{-7,5}{-0,5}\right\}$, это A_4;
- осуществляется пересчёт симплек-таблицы по методу Жордана-Гаусса до получения условия окончания итераций – положительности компонентов A_0.

2.3.2. Решение задачи ЛЦП методом ветвей и границ [7, 41, 50, 63]

Этот метод применяется для решения как полностью целочисленных, так и частично целочисленных задач дискретного программирования.

Пусть математическая модель имеет вид

$$\begin{aligned} &C^T X \to \max, \\ &AX \le B. \end{aligned} \tag{2.29}$$

Компоненты вектора X положительны и целочисленны. Допустим, что исходная задача линейного программирования имеет решение. В этом случае область ограничений замкнута.

Тогда каждая переменная x_j и в допустимом решении, и оптимуме ограничена диапазоном

$$L_j \le x_j \le U_j, \tag{2.30}$$

где L_j – нижний предел, а U_j – верхний предел (граница), которые определяются границами области допустимых решений задачи. Это следует из самого факта наличия непротиворечивых ограничений, образующих замкнутую область

Пусть I есть целое число, такое, что $L_j \le I \le U_j - 1$. Тогда оптимальное ***целочисленное*** значение x_j, удовлетворяющее решению (2.29) и лежащее в пределах (2.30), будет находиться либо между L_j и I, либо между $I + 1$ и U_j. Это приводит к тому, что возникают дополнительные условия, по отношению к исходным условиям (2.29), не противоречащие им:

$$\left.\begin{array}{l} x_j \le I, \\ x_j \ge I+1 \end{array}\right\}. \qquad (2.31)$$

На базе ограничений (2.31) основана систематическая схема применения метода.

Ограничения, приписываемые к исходной задаче, есть ***дополнительные границы***, благодаря чему мы имеем, на каждом шаге постановку ***пары задач*** на базе одной нецелочисленной.

Интерпретация хода решения в виде дерева определило второе название метода – ***ветвей.***

Алгоритм метода ветвей и границ

Композиционно алгоритм состоит из предварительного этапа, проверки условия целочисленности текущего решения, построения задач G_{i1} и G_{i2}, большой итерации, которая представляет собой несколько итерационных шагов двойственного симплекс-метода, и заключительной части, на которой выбирается наилучшее из всех, ранее полученных, целочисленных решений. Цифровой код i в индексации задач соответствует положению текущей "родительской" задачи на дереве решений

1. Предварительный этап. Задача (2.29) решается любым удобным методом до отыскания нецелочисленного оптимального решения, соответствующего точке X_0.

2. Этой точке X_0 ставится в соответствие решение G_0 и его оценка – текущее значение целевой функции $\xi = C^T \times X_0$. Если X_0 – целочисленное решение для основных переменных математической модели, то задача считается решённой.

В противном случае, если X_0 – нецелочисленное решение, то, используя систему неравенств (2.31), получаем множество из двух задач G_{01} и G_{02} (ветвей). Особо подчеркнём, что пара задач ***возникает для одной нецелочисленной переменной одновременно***. Каждая задача решается в отдельности, при этом находят их оценки $\xi(G_{01})$ и $\xi(G_{02})$.

3. В ходе решения на k–й итерации, в зависимости от текущих оценок $\xi(G_{i1})$ и $\xi(G_{i2})$, может произойти дальнейшее ветвление задач.

4. Вычислительный процесс осуществляется до "перерешивания" всех возникающих задач или до получения признаков их неразрешимость. Из полученных решений выбирается то, которое является наилучшим (в смысле оптимума) решением исходной задачи (2.29).

5. Для решения возникающих задач (2.31) используют двойственный симплекс-метод, который, как нам известно, допускает ввод новых ограничений в псевдоплан по ходу решения.

6. Ограничения вводятся только для одной из основных базисных нецелочисленных переменых. Правила формирования ограничений по неравенствам (2.31) суть следующие. Пусть в базисе находится вектор A_j, соответствующая переменная которого x_j – дробное число.

- Задача G_{i1} формируется по ограничению $x_j \le I$, где I – целая часть $[x_j]$, округленная по недостатку. Первоначально формируется ограничение A_{n+t}, которое соответствует канонической форме неравенства и представляется в виде уравнения $I = x_j + x_{n+t}$. В симплекс-таблицу помещается строка, которая получается в результате операции вычитания $A_{n+t} — A_j$.
- Задача G_{i2} формируется по ограничению $- x_j \le - (I + 1)$, которой соответствует каноническая форма $- (I + 1) = - x_j + x_{n+t}$. В симплекс таблицу помещается строка, равная сумме $A_{n+t} + A_j$.

Каждая из исходных симплекс-таблиц задач G_{i1} и G_{i2}, дополняется строкой симплекс-разностей, взятой из таблицы, содержащей нецелочисленное решение G_i.

Продемонстрируем работу алгоритма на известном примере.

Оптимальное решение без учёта целочисленности есть

		c_j	5	6	0	0	0
Базис	$C_Б$	A_0	A_1	A_2	A_3	A_4	A_5
A_2	6	3	0	1	5	–5	0
A_1	5	4,5	1	0	–2,5	7,5	0
A_5	0	0,15	0	0	–0,75	–0,75	1
	δ_j	40,5	0	0	17,5	7,5	0

а его оценка $G_0[40,5] = 40$. Обе задачи формируются для переменной x_1 по 2-й строке таблицы A_2.

Задача G_{01}.

$x_1 \le 4 \Rightarrow A_6$: $4 = x_1 + x_6$, $\widetilde{A}_6 : A_6 - A_1$.

$$
\begin{array}{lrcrcrcrcrcr}
A_6 & 4 & = & 1x_1 & + & 0x_2 & - & 0x_3 & + & 0x_4 & + & 0x_5 & + & 1x_6 \\
- & & & & & & & & & & & & & \\
A_1 & 4{,}5 & = & 1x_1 & + & 0x_2 & - & 2{,}5x_3 & + & 7{,}5x_4 & + & 0x_5 & & 0x_6 \\
\hline
\widetilde{A}_6 & -0{,}5 & = & 0x_1 & + & 0x_2 & + & 2{,}5x_3 & - & 7{,}5x_4 & + & 0x_5 & + & 1x_6
\end{array}
$$

Задача G_{02}.

$-x_1 \le -5 \Rightarrow A_6' : -5 = -x_1 + x_6'$, $\widetilde{A}_6' : A_6' + A_1$

$$
\begin{array}{lrcrcrcrcrcr}
A_6' & -5 & = & -1x_1 & + & 0x_2 & - & 0x_3 & + & 0x_4 & + & 0x_5 & + & 1x_6' \\
- & & & & & & & & & & & & & \\
\hline
A_1 & 4{,}5 & = & 1x_1 & + & 0x_2 & - & 2{,}5x_3 & + & 7{,}5x_4 & + & 0x_5 & & 0x_6 \\
\hline
\widetilde{A}_6' & -0{,}5 & = & 0x_1 & + & 0x_2 & - & 2{,}5x_3 & + & 7{,}5x_4 & + & 0x_5 & + & 1x_6'
\end{array}
$$

Сформируем симплекс-таблицы для обеих задач.

Задача G_{01}.

			c_j	5	6	0	0	0	0
	Базис	$C_Б$	A_0	A_1	A_2	A_3	A_4	A_5	A_6
	A_2	6	3	0	1	5	–5	0	0
	A_1	5	4,5	1	0	–2,5	7,5	0	0
	A_5	0	0,15	0	0	–0,75	–0,75	1	0
←	$\widetilde{A}_6$	0	-0,5	0	0	2,5	–0,75	0	1
		δ_j	40,5	0	0	17,5	7,5	0	0
							↑		

Задача G_{02}.

			c_j	5	6	0	0	0	0
	Базис	$C_Б$	A_0	A_1	A_2	A_3	A_4	A_5	A_6'
	A_2	6	3	0	1	5	–5	0	0
	A_1	5	4,5	1	0	–2,5	7,5	0	0
	A_5	0	0,15	0	0	–0,75	–0,75	1	0
←	$\widetilde{A}_6'$	0	-0,5	0	0	–2,5	–0,75	0	1
		δ_j	40,5	0	0	17,5	7,5	0	0
						↑			

Вид дерева решений показан ниже, на рисунке 2.7.

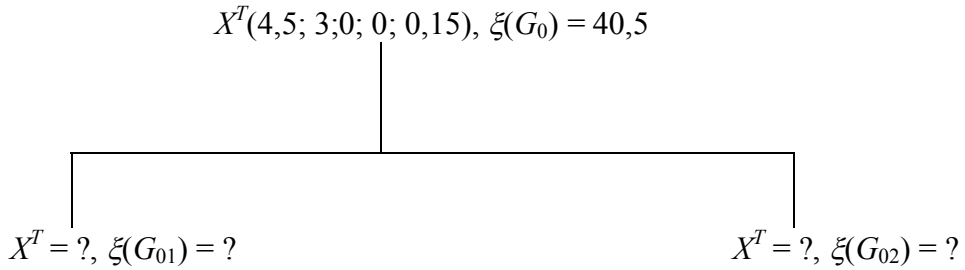

Рисунок 2.7 – Первоначальное дерево решений

Решение опускаем как освоенное нами в предыдущих разделах.

Метод ещё называют ***методом обрыва ветвей*** или ***методом возврата***: всё зависит от способа перемещения по дереву задач. Существует множество алгоритмов метода, адаптированных под разнообразные частные условия содержательной задачи.

2.3.3. Вопросы для самоконтроля

1. В чем сходства и различия терминов “дискретный” и “целочисленный”?
2. Как построить отсекающую плоскость Гомори?
3. Почему алгоритм ветвей и границ получил такое название, что является ветвями, а что — границами?
4. В чём идея сущность и неравенств (2.31)?
5. Почему в ходе решения ЛЦП используется двойственный симплекс-метод?
6. В каких случаях задача ЛЦП не будет иметь решения?
7. Как вы думаете, оптимальное решение ЛЦП будет единственным? Обоснуйте свои соображения по этому поводу.

2.3.4. Транспортные задачи (ТЗ) [7, 8, 22, 27, 33, 33, 65]

2.3.4.1. Постановка ТЗ и общий принцип её решения методом потенциалов

Содержательная постановка транспортной задачи заключается в следующем [22].

Пусть в пунктах $A_1, A_2, \ldots, A_m$, называемыми пунктами производства, в количествах $a_1, a_2, \ldots, a_m$ имеется некоторый однородный продукт.

Указанный продукт потребляется в пунктах $B_1, B_2, \ldots, B_n$, называемых пунктами потребления, в количествах $b_1, b_2, \ldots, b_n$. Стоимость перевозки единицы продукта из пункта A_i в пункт B_j составляет $c_{i,j}$, где $i = 1,m; j = 1,n$.

Решение задачи состоит в нахождении такого плана перевозок $\left\| x_{i,j} \right\|_{m,n} = X$, при котором:

- все запросы потребителя будут удовлетворены;
- весь произведённый товар или продукт будет вывезен из пунктов производства;
- стоимость ***всего объёма*** перевозок будет минимальна.

Условия задачи формулируют в одной из нижеследующих форм.

1. В виде совокупности массивов A, B и C, называемых, соответственно, вектором производства, вектором потребления и матрицей стоимостей.

2. В виде таблицы, комбинирующей указанные выше массивы.

	B_1	B_2	…	B_n	
A_1	$c_{1,1}$	$c_{1,2}$	…	$c_{1,n}$	a_1
A_2	$c_{2,1}$	$c_{2,2}$	…	$c_{i,j}$	a_2
…	...	...	$c_{i,j}$	...	…
A_m	$c_{m,1}$	$c_{m,2}$	…	$c_{m,n}$	a_m
	b_1	b_2	…	b_n	

Обозначив через $x_{i,j}$ количество продукта, перевозимого из i-ого пункта в j-й, можем, руководствуясь благоприобретёнными знаниями, составить математическую модель в терминах линейного программирования

Найти минимум целевой функции

$$L = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{i,j} \cdot x_{i,j}\,, \tag{2.32}$$

при условиях

$$\sum_{i=1}^{m} x_{i,j} \geq b_j\,,\ \ j = \overline{1,n}\ \text{ и} \tag{2.33}$$

$$\sum_{j=1}^{n} x_{i,j} \leq a_i\,,\ \ j = \overline{1,m}\,. \tag{2.34}$$

Ограничение (2.23) говорит о полном удовлетворении спроса потребителей, а (2.34) – о непревышении объёмом перевозок объёма произведенного продукта.

Матрица *X* называется матрицей перевозок, планом перевозок, а тако же планом коммуникаций или коммуникативным планом. Плану перевозок соответствует граф на плоскости, аналогичный представленному на рисунке 2.8 ниже.

Транспортная задача, как особая разновидность задачи линейного программирования, обладает рядом свойств:

- коэффициенты целевой функции неотрицательны (стоимости перевозок не могут быть отрицательными величинами);
- величины элементов вектора производственных запасов и вектора потребления неотрицательны;
- базисное решение закрытой модели транспортной задачи содержит *m+n-1* базисных компонент;
- допустимое решение является планом, а базисное допустимое решение является опорным планом, оптимальное решение – оптимальным планом перевозок.

Дуга графа *[A_i, B_j]* рисунка 2.8 называется коммуникацией, ей соответствует значение $x_{i,j} > 0$ – величина перевозки.

Решение задачи линейного программирования (2.32) – (2.34) известными методами весьма громоздко, причём объём вычислений растёт по мере увеличения *n* и *m*. Поэтому был предложен ряд специальных методов, рассчитанных непосредственно на решение транспортных задач.

Данные методы ориентированы на так называемую “замкнутую” или “закрытую” модель транспортной задачи.

Для такой модели и разрешимости транспортной задачи необходимо и достаточно, чтобы выполнялось ***условие баланса***:

$$\sum_{i=1}^{m} a_i = \sum_{j=1}^{n} b_j\,. \tag{2.35}$$

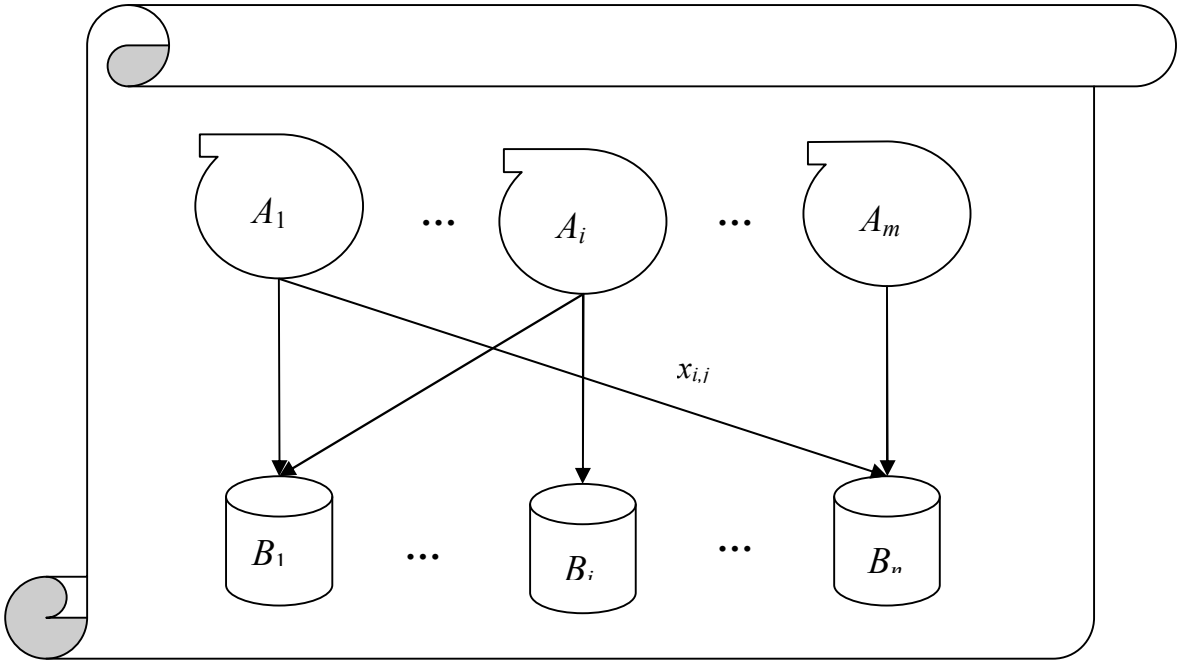

Рисунок 2.8 – Интерпретация плана перевозок

Для задачи с выполняющимся условием баланса в математической модели (2.33) и (2.34) приобретают вид равенства.

Если задача не является сбалансированной (не выполняется условие баланса), то для приведения её к сбалансированному виду необходимо ввести фиктивные пункты производства или пункты потребления. Пункт вводится с таким расчётом, чтобы обеспечить выполнения равенства (2.35), соответствующий компонент вектора A или B равен положительной разнице левой и правой частей (2.35) при отсутствии баланса. Одновременно в матрицу стоимостей C добавляется строка либо столбец с нулевым содержимым.

Иными словами, в случае, когда суммарные запасы превышают суммарные потребности, $\sum_{i=1}^{m} a_i > \sum_{j=1}^{n} b_j$, вводится фиктивный n+1-й потребитель, потребности которого определяются как $b_{n+1} = \sum_{i=1}^{m} a_i - \sum_{j=1}^{n} b_j$. А в случае, когда суммарные потребности превышают суммарные запасы, $\sum_{i=1}^{m} a_i < \sum_{j=1}^{n} b_j$, вводится фиктивный m+1-й поставщик, запасы которого есть $a_{m+1} = \sum_{j=1}^{n} b_j - \sum_{i=1}^{m} a_i$.

Стоимость перевозки единицы груза как до фиктивного потребителя, так и стоимость перевозки единицы груза от фиктивного поставщика полагают равными нулю, так как груз в обоих случаях не перевозится, и, следовательно, не может оказывать влияние на целевую функцию.

Примечание. Математическая модель обладает свойством пропорциональности: как матрицу стоимостей, так и вектора, входящие в условие задачи, можно умножать на коэффициент, отличный от нуля, либо увеличивать (уменьшать) на некоторой число. Это свойство может оказаться полезным для избавления от дробных чисел в условии задачи. Тако же необходимо помнить, что при отрицательных элементах компонентов задача потеряет содержательный смысл.

Исторически первым был подход к решению транспортных задач методом потенциалов. Считается, что данный метод ориентируется на ручное решение задачи, и, в дальнейшем, мы поймём, почему.

Алгоритм метода потенциалов структурно состоит из предварительного и итерационного этапов [29, 33, 34].

Предварительный этап состоит в ***нахождении начального опорного плана*** и расчёта ***потенциалов***.

На итерации

- проверяется условие окончания;
- определяется коммуникация, вводимая в базис;
- определяется коммуникация, выводимая из базиса;
- синхронно пересчитываются план коммуникаций, значения потенциалов и текущее значение функции цели.

Для нахождения начального опорного плана известно несколько методов. Мы рассмотрим ниже ***метод северо-западного угла***, ***метод минимальной стоимости*** и ***метод штрафов***, который часто нахывают по имени автора – методом ***Фогеля***.

Опорный план обладает рядом ***свойств***.

- Он ***невырожден***. Это означает, что число положительных элементов невырожденного плана должно быть ***не менее***, чем ранг матрицы ограничений эквивалентной ЗЛП, то есть

$$r = n + m - 1, \qquad (2.36)$$

где m и n – числа поставщиков и потребителей в ***замкнутой*** модели транспортной задачи.

- Опорный план представляет более или менее удачное решение ТЗ, в котором весь товар вывезён из пунктов производства, а потребности пунктов потребления полностью удовлетворены.
- Из элементов опорного плана нельзя составить замкнутую цепочку.

Для примера будем использовать, в дальнейшем, следующее условие ТЗ

	B_1	B_2	B_3	B_n	
A_1	7	8	1	2	160
A_2	4	5	9	8	140
A_3	9	2	3	6	170
	120	50	190	110,	

на примере которого будем демонстрировать работу алгоритмов.

Проверим условие баланса (2.35):

$$120 + 50 + 190 + 110 = 160 + 140 + 170 = 470,$$

следовательно, мы имеем дело с закрытой математической моделью ТЗ и можем приступать её к решению.

2.3.4.2. Нахождение начального опорного плана ТЗ методом северо-западного угла [33, 34]

Метод получил своё название за “географический” подход к построению плана. По аналогии с топографической картой, левый верхний угол текущего прямоугольного участка плана является северо-западным.

1. Первоначально строится пустая матрица размерами $m \times n$, снизу записываются компоненты вектора производства, а слева – потребления.

2. Для текущего северо-западного угла с координатами (i, j), первоначально (1, 1), выбирается минимальный элемент между i-ым значением вектора производства и j-ым значением вектора потребления

$$x_{i,j} = \min\{a_i, b_j\}, \qquad (2.37)$$

который помещается в план коммуникаций.

3. Элементы векторов производства и потребления, соответствующие заполненной позиции плана, уменьшаются на величину $x_{i,j}$, то есть

$$a_i = a_i - x_{i,j}, \;\; b_j = b_j - x_{i,j}. \qquad (2.38)$$

4. Строка или столбец плана, для которых компоненты векторов производства или потребления обнулились, из рассмотрения исключаются.

Таким образом, возникают координаты нового северо-западного угла: (*i+1, j*), (*i, j+1*) или (*i+1, j+1*).

5. Шаги настоящего алгоритма 2 – 4 продолжаются до тех пор, пока компоненты векторов производства и потребления не окажутся обнулены. Очевидно, что при закрытой модели ТЗ указанное событие будет иметь место.

6. Все элементы плана *X*, не заполненные по завершению работы алгоритма, полагаются равными нулю.

7. Подсчитав число ненулевых элементов, сравниваем его с оценкой (2.36), определяем невырожденность или вырожденность плана.

8. Вычисляем значение целевой функции по формуле (2.28)

$X_0 =$

$120^{1)}$	$40^{2)}$			160	40	0
	$10^{3)}$	$130^{4)}$		140	130	0
		$60^{5)}$	$110^{6)}$	170	110	0
120	50	190	110			
0	10	60	0			
	0	0				

Цифрами со скобками обозначен порядок заполнения плана. Слева и снизу от таблицы показана динамика изменения компонентов a_i и b_j по ходу построения X_0.

Исследуем полученный план на невырожденность: r =3 + 4 – 1 = 6. Число ненулевых элементов плана тако же равно 6-ти. Следовательно, план невырожден.

Рассчитаем значение целевой функции (2.32) для данного плана:

$$L = 7 \times 120 + 8 \times 40 + 5 \times 10 + 9 \times 130 + 3 \times 60 + 6 \times 110 = 3220.$$

Так как расчёты проводились вручную, то формуле (2.32) следовали не буквально, а опуская умножение на ноль.

Замечания.

- Построение опорного плана по методу северо-западного угла осуществляется алгоритмически однозначно.
- Встречается и чисто математическая формулировка алгоритма построения элементов плана

$$\begin{cases} a_\lambda^{(k)} = a_\lambda - \sum_{j=1}^{\mu-1} x_{\lambda,j}, \\ b_\mu^{(k)} = b_\mu - \sum_{i=1}^{\lambda-1} x_{i,\mu}, \\ x_{\lambda,\mu} = \min\left\{a_\lambda^{(k)}, b_\mu^{(k)}\right\}, \end{cases}$$

где литерами (k) обозначен номер итерация.

- Очевидно, что данный способ построения – один из наихудших, ибо ориентируется лишь на координаты заполняемой ячейки плана.

Последнее замечание стимулировало развитие методов, направленных на “удачное” составление начального плана, которые учитывают значения элементов матрицы стоимостей C.

2.3.4.3. Нахождение начального опорного плана ТЗ методом минимальной стоимости [33, 34]

Алгоритм метода опирается на очевидное житейское соображение о целесообразности дешёвой перевозки бОльшего количества товаров. Поэтому построение опорного плана начинается с минимальных элементов по мере возрастания.

1. Так же, как и в предыдущем методе, строится пустая матрица размерами $m \times n$, снизу записываются компоненты вектора производства, а слева – потребления.

2. Элементы матрицы C индексируются в порядке их немонотонного возрастания. Из-за того, что несколько элементов могут иметь одинаковые значения и, как следствие, порядок индексации, очередность заполнения плана, в этом случае, может разниться.

3. Заполнение плана X_0 выполняется в порядке проставленной нумерации, при этом руководствуются выражениями (2.36) и (2.37) для выбора величины перевозки и коррекции векторов A и B.

4. Строка или столбец матрицы стоимостей, для которых компоненты векторов производства или потребления обнулились, из рассмотрения исключаются. Следующая позиция плана определяется минимальным индексом из не исключённой для рассмотрения элементов матрицы C.

5. Шаги алгоритма выполняются до обнуления компонентов векторов A и B. Последнее для замкнутой модели ТЗ неизбежно. Не заполненные элементы плана X полагаются равными нулю. Рассчитывается значение целевой функции и проверяется невырожденность плана.

Замечания.

- Построение опорного плана по методу минимальной стоимости осуществляется алгоритмически неоднозначно. При наличии одинаковых элементов матрицы C, они могут получить разные индексы, в соответствии с волей индексирующего. Как следствие, для одних и тех же условий задачи теоретически могут быть получены различные начальные опорные планы.

- При очевидной целесообразности алгоритма, в процессе работы не учитывается топология распределения минимальных элементов в матрице стоимостей.

Распределения минимальных элементов в матрице *C* позволяет учесть алгоритм метода штрафов, изложение которого ниже.

При ручной реализации алгоритма часто используют таблицу, которая совмещает индексацию, начальный опорный план и матрицу стоимостей “в одном лице”. Пример работы алгоритма представим с использованием совмещённой таблицы.

В левом верхнем углу ячейки таблицы помещается индекс, а в правом верхнем – значение целевой функции, внизу, по центру, помещаются значение величины перевозки.

9) 7	9) 8	1) 1	2) 2	160	0		
		$160^{1)}$					
5) 4	6) 5	11) 9	10) 8	140	20	0	
$120^{4)}$			$20^{6)}$				
12) 9	3) 2	4) 3	7) 6	170	120	90	0
	$50^{2)}$	$30^{3)}$	$90^{5)}$				
120	50	190	110				
0	0	30	20				
		0					

Обратите внимание, что «2» и «8», встреченные в матрице *C*, могли бы иметь и отличную от полученной индексацию.

Мы видим, что сей план не вырожден, а его целевая функция равна

$$L = 160 \times 1 + 120 \times 4 + 20 \times 8 + 50 \times 2 + 30 \times 3 + 90 \times 6 = 1530,$$

что представляет результат, намного лучший, нежели тот, что получен по методу северо-западного угла.

2.3.4.3. Нахождение начального опорного плана ТЗ методом Фогеля [33, 34]

Метод ***Фогеля*** он же ***метод штрафов*** учитывает не только величины элементов матрицы *C*, но и их взаимоположение с другими элементами в столбце либо в строке.

Штрафом вектора, соответствующего строке либо столбцу матрицы C, называется неотрицательная разность между минимальным элементом вектора и следующим за ним ***по величине*** элементом этого же вектора.

Алгоритм метода Фогеля

1. Так же, как и в предыдущих алгоритмах, первоначально строится пустая матрица X_0 размерами $m \times n$, снизу записываются компоненты вектора производства, а слева – потребления.

2. Выполняется расчёт штрафов для столбцов и строк текущей подматрицы матрицы C.

3. Выбирается строка либо столбец с ***максимальным*** штрафом, при этом минимальный компонент для выбранного вектора определит позицию плана, подлежащую заполнению.

4. Заполнение позиции плана X_0 осуществляется согласно выражениям (2.33) и (2.34) для выбора величины перевозки и коррекции векторов A и B.

5. Строки и столбцы матрицы C, соответствующие обнулённым элементам векторов производства и потребления, при определении штрафов ***не рассматриваются***, формируя тем самым новую подматрицу матрицы C, используемую для дальнейших расчётов.

6.Функционирование алгоритма с п. 2 продолжается до тех пор, пока размеры подматрицы матрицы C не сократятся до строки (подстроки) или столбца (подстолбца).

7. Часть плана X_0, соответствующая незаполненной подматрице матрицы C заполняется по правилу минимальной стоимости, в порядке возрастания элементов.

8. Оставшиеся компоненты плана X_0 полагаются равными нулю, рассчитывается целевая функция, план проверяется на невырожденность.

Замечания.

- При равенстве штрафов отсутствует правило выбора для осуществления заполнения, поэтому построение опорного плана осуществляется однозначно.
- Метод Фогеля очень часто (подтверждено опытом) даёт решение ТЗ, совпадающее с оптимальным.

Используем известный численный пример для демонстрации хода выполнения алгоритма:

- литеры $Ш_i$ обозначают штрафы столбцов и строк на i–й итерации;
- серым цветом выделены значение штрафа, используемое на i–й итерации, и заполняемая ячейка плана;

- цифра со скобкой определяет порядок заполнения, который тождественен номеру итерации;
- ячейки (2, 2) и (2, 3) заполняются после четвёртого шага по методу минимальной стоимости.

7	8	1	2	
		$50^{2)}$	$110^{1)}$	160
4	5	9	8	
$120^{4)}$	$20^{6)}$			140
9	2	3	6	
	$30^{5)}$	$140^{3)}$		170
120	50	190	110	
3	3	2	4	Ш$_1$
			0	
3	3	3	–	Ш$_2$
		140		
5	3	6	–	Ш$_3$
		0		
5	3	–	–	Ш$_4$
0				
–		–	–	
	20			
	30			
	0			

Ш$_1$		Ш$_2$		Ш$_3$		Ш$_4$		
1		6		–		–		
	50		0					
1		1		1		1		
							20	0
1		1		1		1		
			0		30		0	

Полученный план является невырожденным. Функция цели есть

$$L = 50 \times 1 + 110 \times 2 + 120 \times 4 + 20 \times 5 + 30 \times 2 + 140 \times 3 = 1330.$$

Этот план обладает самым минимальным значением целевой функции из всех ранее полученных опорных планов.

2.3.4.5. Алгоритм решения ТЗ методом потенциалов

Метод потенциалов является глубокой модификацией симплекс-метода решения ЗЛП применительно к транспортной задаче замкнутого типа. Он позволяет, приняв в качестве начального, некоторое допустимое решение, получить оптимальное решение за конечное число итераций.

Потенциал – это число, характеризующее удобство расположения пункта производства или потребления относительно текущего плана и заданной матрицы стоимостей.

Алгоритм решения транспортной задачи методом потенциалов структурно таков.

1. Проверка условия баланса (2.35) и, при необходимости, балансировка задачи.

2. Предварительный этап, который включает построение начального опорного невырожденного плана и расчёт предварительных потенциалов.

3. Итерация, которая включает в себя проверку на оптимальность решения, определение вводимой в базис коммуникации и коммуникации выводимой из него.

4. Корректировка значения целевой функции и матрицы потенциалов.

5. Структурная схема данного алгоритма представлена ниже, на рисунке 2.9.

Предварительный этап.

На этом этапе необходимо построить опорный план любым методом. План должен быть невырожденным. Для невырожденного опорного плана следует рассчитать потенциалы пунктов производства и пунктов потребления.

Потенциалы пунктов производства и потребления рассчитываются путём решения системы уравнений, которые решаются последовательно для базисных переменных плана.

$$\begin{cases} u_1 = 1, \\ c_{i,j} = v_j - u_i \mid x_{i,j} > 0. \end{cases} \qquad (2.38, \text{а})$$

$$\begin{cases} u_1 = 1, \\ c_{i,j} = v_j + u_i \mid x_{i,j} > 0. \end{cases} \qquad (2.38, \text{б})$$

В формулах (2.38) переменная u_i – потенциалы пунктов производства, а v_j – потенциалы пунктов потребления. Всего имеем $n + m$ неизвестных, а в опорном плане обычно $n + m - 1$ базисных переменных, соответствующих ненулевым коммуникациям.

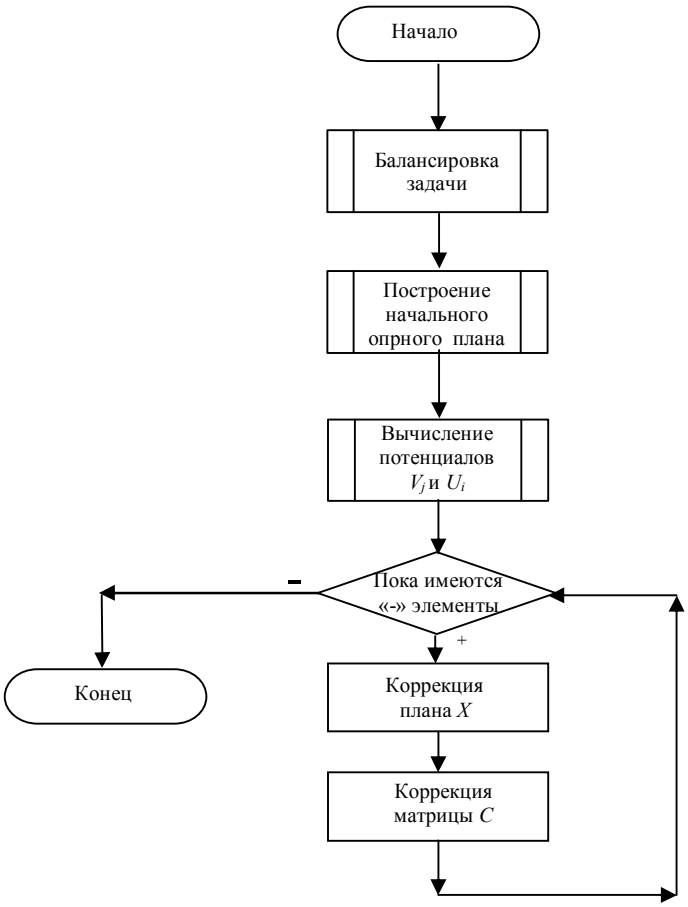

Рисунок 2.9 – Алгоритм метода потенциалов

Поэтому значение потенциала u_1 задаётся принудительно, величина не имеет значения, традиционно это единица или нуль.

Очевидно, что при вырожденном плане уравнений “не хватает”. Для приведения начального плана из вырожденного к невырожденному, в план коммуникаций X_0 добавляются так называемые ε-нули, обозначаемые в плане как 0^{ε} и считающиеся условно положительными. При этом, необходимо, чтобы

- из элементов плана нельзя было составлять замкнутые цепочки и
- обеспечивался в той или иной последовательности обход поворотами на 90° всех элементов плана.

Необходимо добавлять столько ε-нулей, сколько необходимо для получения невырожденного плана.

Постановка ε-нуля задача несколько нетривиальная, как известная головоломка о ферзях под боем. Вручную задача решается достаточно просто, алгоритмическая же её реализация относится к классу “жадных”

комбинаторных задач, требующая для своего решения временных ресурсов, что является минусом метода при реализации.

После расчёта потенциалов пунктов производства и пунктов потребления, матрица стоимости пересчитывается по одной из нижеследующих формул

$$C_{i,j}^{(0)} = c_{i,j} - (v_j - u_i), \qquad (2.39, \text{а})$$

$$C_{i,j}^{(0)} = v_j + u_i - c_{i,j}. \qquad (2.39, \text{б})$$

Причём, должно быть соответствие индексов *а*) и *б*) между формулами определения потенциалов (2.38).

При таком способе расчётов для базисных элементов в матрице C_0 будут получены нули, что может служить для проверки правильности хода вычислений. Матрицу C_0 и последующие C_i можно трактовать как “разность потенциалов” по аналогии с электрическим полем.

Проверка условия оптимальности.

Критерий достижения оптимума формулируется по-разному в зависимости от расчётных формул, применяемых на предварительном этапе:

- для случая а) компоненты текущей матрицы C стоимости должны быть неотрицательны;
- для случая б) компоненты матрицы C неположительные.

Поэтому итерационная часть выполняется до достижения неотрицательности, случай (2.39), а, либо до неположительности, случай (2.39), б, содержимого матрицы C_i.

Итерация.

Определяется коммуникация, вводимая в план X. На неё указывает:

- для случая а) – ***самый отрицательный*** компонент текущей матрицы C,
- для случая б) – ***самый положительный*** компонент текущей матрицы C, который обозначим как δ. Этот элемент называется ***направляющим***.

Соответствующая этой позиции коммуникация будет вводиться в план, обозначают её символом $\mathbf{0}^+$.

Построение кооректирующей цепочки.

Для определения коммуникации, выводимой из базиса, необходимо ***построить замкнутую цепочку***, поворачивающую под углами в 90°. Цепочка в данной задаче играет роль разгрузочного цикла: товар как бы

перемещается в равном количестве по цепочке, где-то прибывая, где-то убывая – закон Ломоносова-Лавуазье для замкнутой модели должен выполняться, ибо опорный план есть решение ТЗ.

Алгоритм построения замкнутой цепочки

Используется метод вычёркивания, суть которого состоит в следующем.

1. Вычеркнуть строки текущего плана X, содержащие менее двух ненулевых элементов. К таковым относятся: собственно коммуникации $x_{i,j}$, коммуникация, вводимая в базис 0^{+} и фиктивные коммуникации 0^{ε}, введённые ранее для обеспечения невырожденности плана.

2. Проделать такую же операцию по столбцам, не учитывая вычеркнутые ранее коммуникации

3. Выполнять пп. 1 и 2 до тех пор, пока удаётся выполнять вычёркивание.

Оставшиеся элементы образуют замкнутую цепочку.

Индексация цепочки.

Начиная с 0^{+}, вводимого в базис, нечётные элементы цепочки отмечаются знаком “–“ «минус», а чётные – знаком “+” «плюс». Направление индексации значения не имеет.

Выбор корректирующего элемента.

Из элементов цепочки выбирается минимальный элемент θ. В выборах участвуют и ε-нули 0^{ε}.

Пересчёт текущего значения целевой функции.

Можно, после коррекции плана коммуникаций, в следующем пункте настоящего алгоритма, воспользоваться формулой (2.32), но существует и более простая расчётная рекурсивная формула

$$L = L - |\delta| \cdot \theta . \qquad (2.40)$$

Коррекция плана коммуникаций.

Построение улучшенного плана коммуникаций осуществляется согласно индексации цепочки: элемент θ прибавляется к элементам, имеющим индекс «плюс» и вычитается из элементов, имеющих знак «минус». Элементы плана X, ***не попавшие*** в цепочку, ***не изменяются***.

Если несколько элементов цепочки, отмеченные знаком «минус» одинаковы и равны θ, то в ходе коррекции план может перестать быть

опорным (базисным). Чтобы этого не произошло ***всем обнулившимся элементам*** цепочки, ***кроме одного***, присваивается значение 0^{ε}.

Назначение ε-нулей 0^{ε} выполняется произвольно, без какого-нибудь критерия. Вовлечение ε-нулей в цепочке делает возможным их использование в качестве корректирующего элемента, таким образом, распределение ε-нулей в матрице X, по мере выполнения итераций, меняется, и становится возможным восстановление первоначальной конфигурации ε-нулей.

Обнаружение этого факта при машинной реализации проблематично – необходимо “помнить” всю последовательность планов коммуникации. Корректировка конфигурации ε-нулей – тако же нетривиальная задача. Поэтому, при машинной реализации алгоритма метода потенциалов поступают следующим образом: вводят параметр, назначение которого – максимальное число итераций, в течение которых целевая функция L неизменна. Превышение числа итераций вызывает остановку вычислительного процесса и позволяет избежать зацикливания [29].

Пересчёт потенциалов и матрицы стоимостей.

В методе потенциалов матрица стоимостей изменяется синхронно с планом коммуникаций. Пересчитать потенциалы и матрицу стоимостей C можно, в принципе, используя формулы (2.35) и (2.36), так сказать, « в лоб», но существует и следующий алгоритм.

Алгоритм преобразования матрицы *C*

1. Отмечаются нули текущей матрицы C, которые соответствуют базисным элементам нового пересчитанного плана коммуникаций X.
2. Вычёркивается направляющая строка.
3. Вычёркиваются столбцы, содержащие вычеркнутые базисные элементы.
4. Вычёркиваются строки, содержащие вычеркнутые при вычёркивании столбцов базисные элементы.
5. Пункты 3 и 4 повторяются циклически, пока элементы матрицы C, соответствующие базисным, окажутся либо не вычеркнуты ни разу, либо вычеркнуты двухкратно.
6. Ко всем элементам вычеркнутых строк необходимо прибавить $|\delta|$, а из всех столбцов – вычесть $|\delta|$ (модуль “дельта”).

Для контроля необходимо отметить, что элементы матрицы C, соответствующие базисным, должны остаться нулевыми.

На этом итерационную часть можно считать законченной.

Демонстрационный пример.
Используем уже известное нам условие.

	B_1	B_2	B_3	B_4	
A_1	*7*	*8*	*1*	*2*	*160*
A_2	*4*	*5*	*9*	*8*	*140*
A_3	*9*	*2*	*3*	*6*	*170*
	120	*50*	*190*	*110*	

Условие баланса для неё выполнено изначально, как мы установили выше, балансировка не потребовалась. Используем в качестве опорного плана невырожеднный план, построенный нами ранее по методу минимальной стоимости.

$$X_0 = \begin{array}{|c|c|c|c|} \hline 0 & 0 & 160^{1)} & 0 \\ \hline 120^{6)} & 0 & 0 & 20^{5)} \\ \hline 0 & 50^{3)} & 30^{2)} & 90^{4)} \\ \hline \end{array} \quad L = 1530$$

Выполним расчёт потенциалов для этого плана, руководствуясь формулой (2.38) и расположением элементов опорного плана.

$u_1 = 1$;
$c_{13} = 1 = v_3 - u_1 = v_3 - 1$; $v_3 = 2$;
$c_{33} = 3 = v_3 - u_3 = 2 - u_3$; $u_3 = -1$;
$c_{32} = 2 = v_2 - u_3 = v_3 + 1$; $v_2 = 1$;
$c_{34} = 6 = v_4 - u_3 = v_4 + 1$; $v_4 = 5$;
$c_{24} = 8 = v_4 - u_2 = 5 - u_2$; $u_2 = -3$;
$c_{21} = 4 = v_1 - u_2 = v_1 + 3$; $v_1 = 1$.

Порядок перемещения по опорному плану в ходе расчёта потенциалов следующий (1, 3) $\Rightarrow$ (3, 3) $\Rightarrow$ (3, 2) $\Rightarrow$ (3,4) $\Rightarrow$ (2, 4) $\Rightarrow$ (2,1), он показан индексами на опорном плане.
Это не единственно возможный порядок расчёта потенциалов. После (3, 3) он мог быть и таким: (3, 3) $\Rightarrow$ (3,4) $\Rightarrow$ (2, 4) $\Rightarrow$ (2, 1), $\Rightarrow$ (3, 2). Важно, чтобы на каждом этапе расчётов в уравнении (2.38) присутствовала только одна неизвестная.

Пересчитаем матрицу C в матрицу C_0, используя формулу (2.39), а.

$c_{11} = 7 - (1 - 1) = 7$; $c_{21} = 4 - (1 + 3) = 0$; $c_{31} = 9 - (1 + 1) = 7$;
$c_{12} = 8 - (1 - 1) = 8$; $c_{22} = 5 - (1 + 3) = 1$; $c_{32} = 2 - (1 + 1) = 0$;
$c_{13} = 3 - (2 - 1) = 0$; $c_{23} = 9 - (2 + 3) = 4$; $c_{33} = 3 - (2 + 1) = 0$;
$c_{14} = 2 - (5 - 1) = -2$; $c_{24} = 8 - (5 + 3) = 0$; $c_{34} = 6 - (5 + 1) = 0$.

Порядок пересчёта здесь не существенен. Видно, что среди элементов матрицы C_0 присутствуют отрицательные, поэтому необходимо улучшать текущий план.

Итерация 1.

Направляющий элемент наибольший отрицательный $c_{1,4} = -2$. Отмечаем в плане X_0 соответствующий элемент как $\mathbf{0}^+$, строим замкнутую цепочку и индексируем её.

$C_0 =$

7	8	$\bar{0}$	-2	x_1
$\bar{0}$	1	4	$\bar{0}$	
7	$\bar{0}$	$\bar{0}$	0	x_3
	x_4	x_2		

$X_0 =$

		160^-	$\mathbf{0}^+$	
120			20	x_3
	50	30^+	90^-	
	x_1	x_2		

Порядок вычёркивания показан крестиками с номерами, вычеркнутые строки и столбцы выделены фоном. Видно, что “белые” элементы образуют цепочку, и больше вычёркиваний произвести не удастся.

Выполняем индексацию от 0^+ хоть влево, хоть вниз, и выбираем элемент для коррекции $\theta = \min\{160,90) = 90$. Текущее значение функции цели при этом станет, согласно (2.40)

$$L = 1530 - |-2| \times 90 = 1350.$$

Соответствующий этой функции план X_1, полученный после коррекции, показан ниже. Пересчитаем матрицу C. Обозначим «крышками» сверху нули, ***соответствующие базисным элементам нового плана*** (важно!!!). Порядок вычёркивания будем обозначать крестиком с цифрой – номером вычёркивания, а вычеркнутые строки и столбцы обозначать фоном.

Прибавляем $|\delta| = 2$ к вычеркнутым строкам и вычитаем от вычеркнутых столбцов. Очевидно, что, в ходе этой операции, дважды вычеркнутые и ни разу не вычеркнутые элементы останутся неизменными.

$C_1 =$

9	8	0	0
0	-1	2	0
9	0	0	2

$X_1 =$

		70	90
120			20
	50	120	

Так как среди элементов матрицы C_1 имеются отрицательные числа, то решение не закончено.

Итерация 2.

$C_1 =$

9	8	$\bar{0}$	$\bar{0}$
$\bar{0}$	– 1	2	0
9	$\bar{0}$	$\bar{0}$	2

x_2

x_1 $X_1 =$

		$70^{-}\rightarrow$	$90^{+}\downarrow$
120	$\mathbf{0}^{+}\downarrow$	$\leftarrow\uparrow$	$\leftarrow 20^{-}$
	$50^{-}\rightarrow$	$120^{+}\uparrow$	

x_1

Направляющий элемент в C_1 имеет координаты (2,2). Помещаем $\mathbf{0}^{+}$ в соответствующую позицию плана X_1 и приступаем к вычёркиванию. Получающаяся при этом цепочка имеет вид неправильной восьмёрки, проход по которой показан стрелками.

Индексация позволяет выбрать направляющий элемент $\theta = \min\{50,70,20\} = 20$. Целевая функция при этом составит

$$L = 1350 - |-1| \times 20 = 1330,$$

а соответствующий скорректированный план X_2 показан ниже. Пересчитаем теперь матрицу C_1, предварительно отметив базисные элементы плана X_2 в C_1.

Вычеркнем 2-ю строку и 1-й столбец, вычитая и прибавляя *1*, получим матрицу C_2, которая с соответствующим планом X_2 показана ниже.

$C_2 =$

8	8	0	0
0	0	3	1
8	0	0	2

$X_2 =$

		50	110
120	20		
	30	140	

Полученный план совместно со значением целевой функции, есть оптимальное решение транспортной задачи. Сравнив полученный результат с опорным планом, построенным по методу Фогеля, заметим, что оба решения полностью совпадают.

2.3.3.6. Алгоритм решения ТЗ венгерским методом [22, 29]

Данный метод был предложен в 1931 г. Р. Эгервари, венгром по национальности, и 1953 г. подвёргся модификации Г. Куном. В честь родины авторов получил своё наименование. Точнее, этим названием

обозначается группа методов. Задача представляется общепринятой замкнутой моделью (2.32) – (2.34).

Метод базируется на ряде определений.

1. ***Суммарной невязкой плана*** называют величину, определяемую выражением

$$\Delta = \sum_{i=1}^{m} a_i + \sum_{j=1}^{n} b_j - 2 \cdot \sum_{i=1}^{m} \sum_{j=1}^{n} x_{i,j} \,. \tag{2.40}$$

2. Величины

$$\delta_j = b_j - \sum_{i=1}^{m} x_{i,j} \tag{2.41}$$

и

$$\delta_i = a_i - \sum_{i=1}^{m} x_{i,j} \tag{2.42}$$

называют, соответственно, ***невязками по столбцам*** и ***невязками по строкам***. Поддаётся выводу и следующее выражение

$$\Delta = \sum_{i=1}^{m} \delta_i + \sum_{j=1}^{n} \delta_j \,. \tag{2.43}$$

3. Столбцы и строки матрицы C, отмеченные в ходе работы алгоритма символом «плюс» называются ***выделенными***.

4. Нулевой элемент матрицы C ($c_{i,j} = 0$), для которого в плане X существует коммуникация ($x_{i,j} \geq 0$), называется ***существенным нулём*** матрицы C.

Алгоритм венгерского метода

Граф-схема алгоритма представлена на рисунке 2.10. Структурно алгоритм состоит из следующих пунктов.

1. Предварительный этап.
2. Проверка оптимальности полученного плана.
3. Итерационная часть.

В свою очередь, итерационная часть состоит из трёх этапов, которым предшествует этап разметки:

- 1-й этап – поисковый;
- 2-й этап – построение цепочки и коррекция плана;

- 3-й этап – выполнение эквивалентных преобразований матрицы *C*.

Итерация, после разметки, начинается первым этапом, а оканчивается вторым, причём в процессе одной и той же итерации первый и третий этапы могут многократно повторяться.

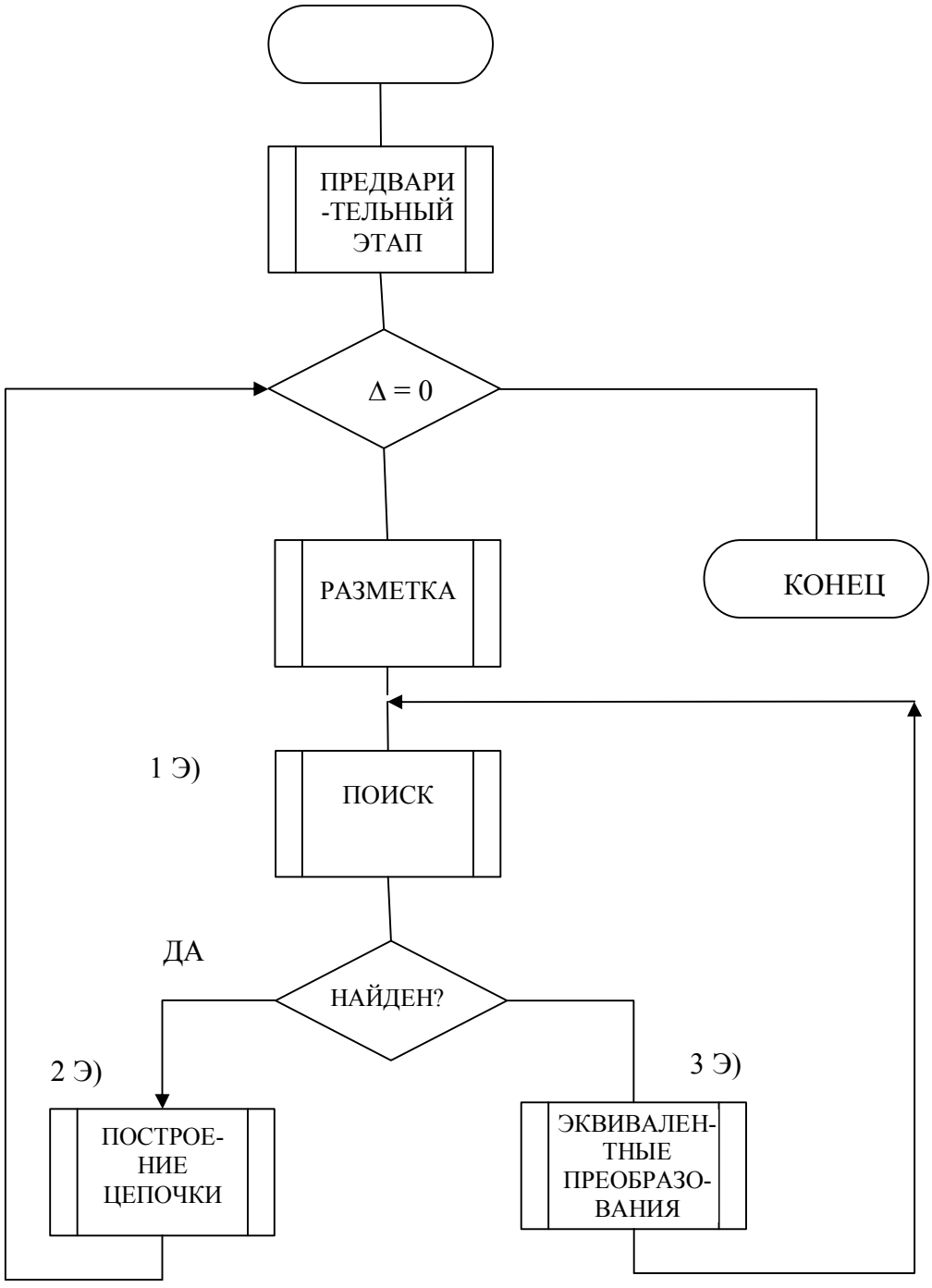

Рисунок 2.10 – Алгоритм венгерского метода

Предварительный этап.

Состоит в построении начального плана X_0 и определении его невязок.

- В каждом столбце матрицы C отыскивается минимальный элемент, который затем вычитается из всех элементов этого столбца. В результате, матрица C преобразуется в матрицу $C`$.
- Проделав аналогичную операцию над строками в матрице $C`$, получим матрицу C_0.
- Для нулевых элементов матрицы C_0, перемещаясь по столбцам сверху вниз и слева направо, заполним план X_0. Порядок определения коммуникаций, коррекции векторов производства и потребления рассмотрен нами ранее при построении начальных опорных планов.
- Вычисляется суммарная невязка (2.40) полученного плана.

Заметим, что невязки по столбцам (2.41) и строкам (2.42) получаются автоматически в ходе построения начального плана, а суммарную невязку проще получить, используя (2.43).

В отличие от опорных планов метода потенциалов, планы венгерского метода таковыми не являются (смотри свойства опорных планов).

Проверка условия окончания.

Если суммарная невязка Δ текущего плана X равна нулю, то полученный план является оптимальным. Необходимо рассчитать целевую функцию (2.32).

ИТЕРАЦИОННАЯ ЧАСТЬ АЛГОРИТМА.

Разметка.

Разметка текущей матрицы C выполняется в начале итерации и сохраняется до её конца с теми изменениями, которые вносятся в неё по мере выполнения алгоритма.

- Выделить знаком «плюс» j-е столбцы матрицы C, обладающие нулевой невязкой $\delta_j = 0$.
- Выделить чертой сверху существенные нули матрицы C.

Этап 1 – этап поиска.

Область поиска: невыделенная часть матрицы C – невыделенные столбцы и строки.

Цель поиска: найти в невыделенной части матрицы C нуль, стоящий в строке, которой в плане X соответствует положительная невязка $\delta_i \geq 0$.

Поисковый этап заканчивается одним из случаев:

- если все нули матрицы C находятся в выделенной части, то необходимо перейти к этапу 3 – эквивалентных преобразований матрицы C;
- поиск завершился успешно, найден ноль в строке с положительной невязкой. В этом случае далее выполняется этап 2 – построение цепочки и коррекция плана коммуникаций.

В ходе поиска невыделенная часть матрицы C просматривается по столбцам сверху вниз а столбцы – слева направо (“По-китайски”).

Пусть среди элементов найден ноль. Его отмечают апострофом (штрихом) и анализируют невязку по строке δ_i.

Если невязка δ_i положительна, то этот ноль со штрихом является искомым, а поиск заканчивается успешно.

Если невязка δ_i нулевая, то текущая стока выделяется знаком «плюс», и просматривается по местам её пересечения с выделенными столбцами. Если в месте пересечения стоит существенный нуль, то его обозначают звёздочкой (*), а знак выделения над столбцом уничтожают, обводя кружком или заключая в скобки. Столбец становится невыделенным и делается доступным для поиска. Поиск далее продолжают по этому столбцу со снятым выделением.

Этап 2 – этап построения цепочки и коррекции плана.

Цепочка ***не замкнута***, так как существуют невязки, которые, в замкнутой модели, фактически означают наличие неудовлетворённого спроса и невывезенного товара. За счёт последних и будет пополняться план коммуникаций.

1. Цепочка составляется из нулей со штрихом (0’) и нулей со звёздочками (0^*), содержит нечётное число элементов, и, в принципе, может состоять и из одного нуля со штрихом.

2. Цепочка начинается от последнего найденного нуля со штрихом 0’ к нулю со звездой 0^* по столбцу, далее, по направлению под 90° к предыдущему, по строке к нулю со штрихом и так далее. На нечётных местах цепочки будут стоять нули со штрихом, а на чётных – нули со звёздами. Цепочка начинается в строке с положительной невязкой и заканчивается в столбце с положительной невязкой нулём со штрихом.

3. Выбирается корректирующий элемент

$$\theta = \min\left\{\delta_i^{НАЧАЛА}, \delta_j^{КОНЦА}, x_{i,j}^*\right\}, \qquad (2.44)$$

где $\delta_i^{НАЧАЛА}$ – невязка строки начала цепочки; $\delta_j^{КОНЦА}$ – невязка столбца конца цепочки; $x_{i,j}^*$ – элементы, стоящие на чётных местах цепочки.

4. После выбора θ текущий план преобразуется по алгоритму:

$$x_{i,j} = \begin{cases} x_{i,j}, & \text{не входит в цепочку,} \\ x_{i,j}^{\nabla} + \theta, & \text{нечётный элемент цепочки,} \\ x_{i,j}^{*} - \theta, & \text{чётный элемент цепочки.} \end{cases} \qquad (2.45)$$

5. Рассчитываются Δ, δ_j и δ_j по соответствующим формулам, удобным рассчитывающему (2.40) – (2.43).

Этап 3 – этап эквивалентного преобразования матрицы C.

1. Среди невыделенных элементов матрицы C выбирается минимальный положительный (а другой и быть не может, но так – в первоисточнике) элемент $h > 0$. Этот элемент называется ***корректирующим***.

2. Корректирующий элемент вычитается от невыделенных строк матрицы C.

3. Корректирующий элемент прибавляется к выделенным столбцам матрицы C.

При этом, очевидно, выделенные однократно (сиречь, стоящие в выделенной строке либо в выделенном столбце) элементы не изменятся, двукратно выделенные – увеличатся на величину h, а в невыделенной части матрицы появится хотя бы один нуль, который, в дальнейшем, буде обработан алгоритмом поиска на 1-м этапе.

Замечания по методу Эгервари.

1. В ходе работы алгоритма не используются опорные планы, поэтому зацикливание при машинной реализации не возникает.

2. По величине суммарной невязки Δ можно грубо (наихудший из возможных ход решения) оценить число итераций до получения оптимального решения:

$$N_{ОСТ} \le \frac{\Delta}{2}.$$

При получения последней формулы учтено, что при записи в план коммуникаций некоторого числа $к$ суммарная невязка текущего плана Δ уменьшается на величину, равную $2 \times к$.

Предположив, что за каждую итерацию план коммуникаций помещается единица товара, придём к обсуждаемой формуле.

Демонстрационный пример.
Решить транспортную задачу:

	B_1	B_2	B_3	B_n	
A_1	7	8	1	2	160
A_2	4	5	9	8	140
A_3	9	2	3	6	170
	120	50	190	110	

Задача сбалансирована, модель замкнутая.

Предварительный этап.
Минимальные элементы в столбцах матрицы C показаны фоном. Их вычитание позволяет получить матрицу C'.

$C' =$

3	6	0	0
0	3	8	6
5	0	2	4

$C' = C_0 =$

3	6	0	0
0	3	8	6
5	0	2	4

Так как в каждой строчке есть по минимальному элементу – нулю (серый фон), то матрицы C' и C_0 совпадут.
Для данной матрицы C_0 по её нулям строим начальный план X_0, показанный ниже справа.

$C_0 =$

+	+		
3	6	$\bar{0}$	0
$\bar{0}$	3	8	6
5	$\bar{0}$	2	4

+

$X_0 =$

				δ_i
		160		0
120				20
	50			120
0	0	30	110	δ_j

Суммарная невязка этого плана, согласно (2.43) составляет

$$\Delta = 30 + 110 + 20 + 120 = 280,$$

поэтому итерационная часть алгоритма неизбежна.

1-я итерация.
Все почти нули матрицы существенные, обозначим их как $\bar{0}$. Выделим также 1-й и 2-й столбцы как имеющие нулевые невязки.

1-й этап. Поиск.

Отмечаем первый встреченный невыделенный нуль с координатами (1, 3) штрихом. Его невязка по строке – нулевая, поэтому строка отмечается плюсом. На этой строке нет существенных нулей, стоящих в выделенных столбцах.

В невыделенной части матрицы (серая) нет нулей, поэтому этап поиска закончился неудачей, необходимо осуществлять эквивалентные преобразования матрицы C_0.

3-й этап.

В невыделенной (серой) части матрицы минимальный элемент $h = 2$. Прибавим и отнимем его согласно разметке. Получим матрицу C_1, отметим, что вся индексация переносится. В её невыделенной части образовался нуль, снова переходим к этапу поиска.

Поиск.

Отмечаем штрихом нуль с координатами (3, 3). Его построчная невязка $\delta_j = 120$ – положительна, следовательно, этап поиска завершился успешно. Можно строить цепочку.

2-й этап.

Цепочка состоит из одного элемента. Выберем элемент коррекции:

$$\theta = \min \{30, 120\} = 30.$$

Скорректируем план и невязки:

$$\Delta = \Delta - 2 \times \theta = 280 - 60 = 220.$$

Невязка положительна, решение не закончено. План X_1 представлен ниже

$C_1 =$

+	+			
5	*8*	$\bar{0}$`	0	+
$\bar{0}$	*3*	*6*	*4*	
5	$\bar{0}$	*0*`	*2*	

$X_1 =$

				δ_i
		160		0
120				20
	50	30		90
δ_j 0	0	0	110	

2-я итерация.

Размечаем матрицу C_1 и приступаем к поиску.

1-й этап, поисковый.

Выделим штрихом нуль с координатами (1, 4). Он стоит в строке с нулевой невязкой, поэтому строку выделяем плюсом. Просматриваем её пересечения с выделенными столбцами.

$C_1 =$

+	+	(+)	
5	*8*	$\bar{0}^* \rightarrow$	$\rightarrow 0`$
$\bar{0}$	*3*	$\uparrow 6$	*4*
5	$\bar{0}$	$\uparrow \bar{0}`$	*2*

+

$X_1 =$

				δ_i
		160		*0*
120				*20*
	50	30		*90*
0	*0*	*0*	*110*	
δ_j				

На пересечении с выделенным столбцом – существенный нуль, отмечаем его звёздочкой, снимаем выделение столбца, заключая его в скобки. Продолжаем поиск в третьем столбце, отмечаем нуль (3, 3) штрихом. Его невязка положительна, этот ноль – искомый, этап поиска закончился удачно.

2-й этап.

Цепочка из элементов матрицы C_1 есть последовательность (3, 3) $\Rightarrow$ (1, 3) $\Rightarrow$ (1, 4) выделена в матрице серым. Выберем корректирующий элемент: невязка строки начала – 90, невязка столбца конца цепочки – 110, $X(\bar{0}^*) =$ 160.

$$\theta = \min \{90, 110, 160\} = 90.$$

Выполним изменение элементов, соответствующих цепочке в плане X_1, преобразуя его в X_2.

Суммарная невязка этого плана составит

$$\Delta = 220 - 2 \times 90 = 40.$$

Необходимо провести очередную итерацию.

$X_2 =$

				δ_i
		70	90	0
120				20
	50	120		0
0	0	0	20	
δ_j				

3-я итерация.

Размечаем матрицу C_1 и приступаем к поиску.

1-й этап, поисковый.

Выделим штрихом нуль с координатами (1, 4). Он стоит в строке с нулевой невязкой, поэтому строку выделяем плюсом. Просматриваем её пересечения с выделенными столбцами.

На пересечении с выделенным столбцом – существенный нуль, отмечаем его звёздочкой, снимаем выделение столбца, заключая его в скобки. Продолжаем поиск в третьем столбце, отмечаем нуль (3, 3) штрихом. Его невязка тако же нулевая, отмечаем строку плюсом, просматриваем пересечения с выделенными столбцами.

По результатам просмотра нуль (3, 2) получает звезду, а с третьего столбца – снимается выделение. Больше в невыделенной части нулей нет. На этом поиск заканчивается неудачей.

$C_1 =$

	+	(+)	(+)	
	5	8	$\bar{0}^*$	$\bar{0}$`
+	$\bar{0}$	3	6	4
+	5	$\bar{0}^*$	$\bar{0}$`	2

$X_2 =$

				δ_i
		70	90	0
120				20
	50	120		0
0	0	0	20	

3-й этап, эквивалентные преобразования матрицы C_1.

В невыделенной части матрицы C_1 (выделена серым, извиняюсь за каламбур) отыскиваем положительный элемент для коррекции $h = 3$. В результате имеем матрицу C_2, показанную ниже.

$C_2 =$

	+			
	8	8	$\bar{0}^* \rightarrow$	$\rightarrow \bar{0}$`
+	$\bar{0}$	$\downarrow 0$`	$\uparrow 3$	1
+	8	$\bar{0}^* \rightarrow$	$\uparrow \bar{0}$`	2

$X_2 =$

				δ_i
		50	110	0
120	20			0
	30	140		0
0	0	0	0	

Приступаем к поиску, 1-й этап.

Отмечаем штрихом нуль (2, 2). Его невязка по строке показывает, что этот нуль является искомым.

Этап 2-й.

Строим цепочку (2, 2)` $\Rightarrow$ (3, 2)$^*$ $\Rightarrow$ (3, 3)` $\Rightarrow$ (1, 3)* $\Rightarrow$ (1, 4)`, она показана фоном и стрелками. Корректирующий элемент

$$\Delta = \min\{\delta_2 = 20,\ \delta_4 = 20,\ x_{3,2} = 50,\ x_{13} = 70\} = 20.$$

Полученный план перевозок X_2 показан рядом с матрицей C_2 его невязка нулевая.

Следовательно, достигнут оптимум, можно рассчитать целевую функцию $L = 1330$.

Это столько же, как и в методе потенциалов. Планы перевозок, полученных обоими методами, совпадают.

Замечания по процессу решения.

1. Текущие матрицы C и X изменяются несинхронно. В принципе, возможны решения, когда в ходе нескольких итераций матрица C остаётся одним и тем же, а план X – неоднократно меняется.

2. Известны случаи, когда при равенстве целевых функций, при одном и том же условии, решённом венгерским методом и методом потенциалов, получаются разные планы перевозок. Это есть своеобразная иллюстрация теорем линейного программирования (смотри раздел ранее).

2.3.4.7. Алгоритм решения ТЗ с ограниченной пропускной способностью коммуникаций [20, 24, 28 – 30]

Общая постановка транспортной задачи, ранее нами рассмотренная, характерна тем, что возможность перевозки $x_{i,j}$ единиц товара от i-го поставщика j-му потребителю физически ничем не ограничивается. Однако повседневная практика перевозок может повлечь за собой ограничения на количество перевозимого груза.

Указанные ограничения могут иметь различную физическую или иную природу:

- быть связанными с затратами на горючее (лимитироваться);
- определяться грузоподъёмностью транспортных средств, их проходимостью или предельным числом рейсов;
- диктоваться соотношением полезного груза в функции от дистанции перевозки, как это бывает, например, в дальнебомбардировочной авиации;
- и т.п.

  В таких случаях говорят о транспортной задаче с ограниченными пропускными способностями коммуникаций. Её математическая модель несколько отличается от общеизвестной (2.32) – (2.34), в части дополнительных ограничений:

$$\min \sum_{i=1}^{m}\sum_{j=1}^{n} c_{i,j} \cdot x_{i,j};$$

$$\sum_{j=1}^{m} x_{i,j} = a_i; \sum_{i=1}^{n} x_{i,j} = b_j;$$

$$0 \le x_{i,j} \le d_{i,j}; \qquad i = 1,\overline{m}; j = 1,\overline{n}.$$

Величина $d_{i,j}$ называется ограничением коммуникации, благодаря этому задачу в данной постановке называют ещё *Td*-задачей, т.к. подразумевается, что в дополнение к известным векторам и матрицам *A*, *B* и *C* добавляется еще и матрица ограничений *D*.

Td-задачи решаются с помощью модифицированного алгоритма венгерского метода, модификации касаются учёта ограничений в ходе решения.

При изложении метода будем использовать следующие определения.

1. Элемент $c_{i,j} = 0$ матрицы *C* называется ***X-неполным нулём*** (или просто ***неполным нулём***), если в плане *X* решаемой *Td*-задачи величина $x_{i,j} < d_{i,j}$ (меньше пропускной способности коммуникаций).

2. ***Полным нулём*** (***X-полным нулём***) называется элемент $c_{i,j} = 0$ матрицы *C*, для которого $x_{i,j} = d_{i,j}$ (равен пропускной способности коммуникации).

3. Элемент $c_{i,j} = 0$ матрицы *C* является ***существенным нулём***, если $x_{i,j} > 0$ в плане перевозок *X*.

4. Если $x_{i,j}$ в плане перевозок *X* нулевой, то соответствующий нуль матрицы $c_{i,j}$ называется ***несущественным нулём***.

5. Элемент, находящийся в матрице *C* на пересечении выделенной строки и выделенного столбца, называется ***дважды выделенным***.

Функционирование алгоритма

Венгерский алгоритм решения *Td*-задачи имеет такую же структуру, показанную на рисунке 2.9, как и при отсутствии ограничений с точностью до названия этапов. Поэтому алгоритм дадим в отличиях и дополнениях.

Предварительный этап.

Заполнение плана *X* осуществляется не на основании формулы (2.37), а с учётом ограничений:

$$X_{i,j} = \min\{a_i,\ b_j,\ d_{i,j}\}. \qquad (2.46)$$

ИТЕРАЦИЯ.

Разметка.

Дополнительно отмечают ***существенные нули*** матрицы *C*: точкой сверху – *x*-неполные нули $\dot{0}$, двумя точками – x-полные $\ddot{0}$.

1-й этап, поисковый.

Цель поиска: найти ***неполный нуль*** матрицы *C*, независимо от того существенный он или несущественный.

2-й этап, построение цепочки и коррекция плана X.

Корректирующий элемент выбирается по правилу

$$\theta = \min\left\{\delta_i^{НАЧАЛА}, \delta_j^{КОНЦА}, x_{i,j}^{*}, r'_{i,j}\right\}, \qquad (2.47)$$

где $\delta_i^{НАЧАЛА}$ – невязка строки начала цепочки; $\delta_j^{КОНЦА}$ – невязка столбца конца цепочки; $x_{i,j}^{*}$ – элементы, стоящие на чётных местах цепочки, $r'_{i,j} = d_{i,j} - x_{i,j}$ – величина насыщения, необходимая для приведения коммуникации, стоящей на нечётной позиции в цепочке, к *x*-полному нулю.

3-й этап, коррекция матрицы *C*.

1. Корректирующий элемент *h* определяется как минимальный среди ***невыделенных положительных*** элементов, и, взятых ***по модулю***, ***дважды выделенных отрицательных***.

$$h = \min\{c_{i,j} > 0, |c_{i,j}^{+} < 0|+\}. \qquad (2.48)$$

2. Если все *x*-неполные нули выделены при нулевых невязках, все невыделенные элементы *C* отрицательны, а дважды выделенные – положительны, то *Td*-***задача неразрешима***.

3. Если в процессе коррекции ***дважды выделенный отрицательный*** элемент матрицы *C* ***становится нулём***, то его ***помечают звёздочкой***, а ***знак выделения*** над столбцом ***уничтожают***, делая столбец доступным для выполнения поисковых операций.

Пример решения задачи

Используем условие уже известного примера, дополненное ограничениями.

	B_1	B_2	B_3	B_4	
A_1	7	8	1	2	160
A_2	4	5	9	8	140
A_3	9	2	3	6	170
	120	50	190	110	

$D =$

50	100	200	50
150	50	100	50
50	50	50	50

Условие сбалансировано, модель замкнутая.

Предварительный этап.

На предварительном этапе матрица C_0 будет такая же, как и в случае ТЗ без ограничений.

$C_0 =$

+	+		
3	6	$\dot{0}^{\nabla}$	0
$\dot{0}$	3	8	6
5	$\ddot{0}$	2	4

$+$ $X_0 =$

				δ_i
		160		0
120				20
	50			120
0	0	30	110	

δ_j

Начальный план X_0 получился тоже идентичный, поскольку он (2.46) соответствует. Заполнение X_0 читателю рекомендуется проделать самостоятельно. Суммарная невязка плана $\Delta = 280$.

1-я итерация.

Этап разметки.

Нули (1, 3), (2, 1) являются неполными, нуль (3, 2) – полным, выделяем 1-й и 2-й столбцы.

1-й этап.

Нуль (1, 3) выделяем штрихом, а первую строку – плюсом, так как её соответствует нулевая невязка в плане X_0. На этом поиск закончен, необходимы эквивалентные преобразования.

3-й этап.

Корректирующий элемент в невыделенной, серой части матрицы, согласно (2.46), равен 2. Применение алгоритма коррекции даёт C_1.

$C_1 =$

+	+			
5	8	$\dot{0}^{\nabla}$	0	+
$\dot{0}$	3	6	4	
5	$\ddot{0}$	0^{∇}	2	

$X_1 =$

				δ_i
		160		0
120				20
	50	30		90
0	0	0	110	

δ_j

1-й этап.

Выделяем элемент (3, 3) штрихом, на этом этап поиска заканчивается удачно.

2-й этап.

Цепочка состоит из одного элемента. Соответствующую позицию плана X заполняем элементом $\theta = \min\{120, 30, 50\} = 30$. $r'_{3,3} = d_{3,3} - x_{33} = d_{3,3} - 0 = 50$. Получаем план коммуникаций X_1.

Невязка полученного плана $\Delta = 220$, расчёты продолжаются.

2-я итерация.

Этап разметки.

$C_1 =$

+	+	(+)	
5	8	$\dot{0}^{*}$	$\rightarrow 0^{\nabla}$
$\dot{0}$	3	$\uparrow 6$	4
5	$\ddot{0}$	$\uparrow \dot{0}^{\nabla}$	2

$X_2 =$

				δ_i
		140	20	0
120				20
	50	50		70
0	0	0	90	

δ_j

Нуль, добавленный на предыдущей итерации, неполный, столбцы с первого по третий – обладают нулевыми невязками в плане X_1, выделяем их соответствующим образом.

1-й этап.

Нуль (1, 4) выделяем штрихом, а первую строку – плюсом, поскольку её невязка нулевая. Нуль (1,3) выделяем звёздочкой, снимаем выделение с третьего столбца, продолжаем поиск. Нуль (3, 3) выделяем штрихом, этот нуль – искомый, будем строить цепочку.

2-й этап.

Цепочка соединяет нули с координатами в матрице C_1 $(3, 3)^{\nabla} \Rightarrow (1, 3)^{*} \Rightarrow (1, 4)^{\nabla}$.

Выбираем корректирующий элемент по формуле (2.47)

$$\theta = \min\{\delta_3 = 90, \delta_4 = 110, x^*_{1,3} = 160, r'_{3,3} = 20, r'_{3,3} = 50\} = 20$$

и перестраиваем план. При этом невязка получится Δ = 180. Итерации продолжаются.

3-я итерация.

Этап разметки.

$C_1 =$

+	+	(+)		
5	*8*	0̇ *	0̇ ∇	+
0̇	*3*	*6*	*4*	
5	0̈	0̈	*2*	

$X_2 =$

				δ_i
		140	20	*0*
120				*20*
	50	50		*70*
0	*0*	*0*	*90*	

δ_j

Нули (1, 3), (1, 4) и (2, 1) являются неполными, нули (3, 2) и (3, 3) – полными, выделяем столбцы с 1-го по 3-й.

1-й этап.

Нуль (1, 4) выделяем штрихом, а первую строку – плюсом. Нуль (1,3) выделяем звёздочкой, снимаем выделение с третьего столбца, продолжаем поиск. Больше неполных нулей в невыделенной части матрицы нет.

3-й этап.

Корректирующий элемент матрицы h = 2. Преобразуем матрицу C_1 в C_2.

1-й этап.

Отмечаем нуль (3, 4) штрихом. Его невязка положительна, он несущественный и неполный. Поиск завершён.

3-й этап.

Цепочка состоит из одного нуля с координатами (3, 4), эта позиция в плане заполняется исходя из невязок строки, столбца и ограничения коммуникации

$$\theta = \min\{\delta_3 = 70, \delta_4 = 90, r'_{3,4} = 50\} = 50.$$

Получим план X_3. Его невязка Δ = *80*. Следовательно, оптимум не получен.

$C_2 =$

+	+			
7	10	$\dot{0}^{*}$	$\dot{0}^{\nabla}$	+
$\dot{0}$	3	4	2	
5	$\ddot{0}$	–2	0^{∇}	

$X_3 =$

				δ_i
		140	20	0
120				20
	50	50	50	20
0	0	0	40	

δ_j

4-я итерация.

Этап разметки.

$C_2 =$

+	+	(+)		
7	10	$\dot{0}^{*}$	$\dot{0}^{\nabla}$	+
$\dot{0}$	3	4	2	
5	$\ddot{0}$	–2	$\ddot{0}$	

$X_3 =$

				δ_i
		140	20	0
120				20
	50	50	50	20
0	0	0	40	

δ_j

Нули (1, 3), (1, 4) и (2, 1) являются неполными, нули (3, 2) и (3, 4) – полными, выделяем столбцы с 1-го по 3-й.

1-й этап.

Нуль (1, 4) выделяем штрихом, а первую строку – плюсом. Нуль (1,3) выделяем звёздочкой, снимаем выделение с третьего столбца, продолжаем поиск. Больше неполных нулей в невыделенной части матрицы нет.

3-й этап.

Корректирующий элемент матрицы $h = 2$ выбираем из невыделенной серой части. Преобразуем матрицу C_2 в C_3.

$C_3 =$

+	+			
9	*12*	$\dot{0}^{*}$	$\dot{0}^{\nabla}$	+
$\dot{0}$	*3*	*2*	0^{∇}	
5	$\ddot{0}$	*–4*	*–2*	

$X_4 =$

				δ_i
		140	20	*0*
120			20	*0*
	50	50	50	*20*
0	*0*	*0*	*20*	

δ_j

1-й этап.

Отмечаем штрихом нуль (2, 4), он стоит в строке с положительной невязкой, и неполный несущественный. Поиск закончен успешно.

2-й этап.

Имеем цепочку и одного нуля, помещаем в позицию (2, 4) значение

$$\theta = \min\{\delta_2 = 20, \delta_4 = 40, r'_{2,4} = 50\} = 20 .$$

Получаем план X_4. Его суммарная невязка $\Delta = 40$. Итерации продолжаются.

5-я итерация.

Этап разметки.

$C_3 =$

(+)	+	(+)		
9	12	$\dot{0}^{*}$	$\dot{0}^{\nabla}$	+
$\dot{0}^{*}$	3	2	$\dot{0}^{\nabla}$	+
5	$\ddot{0}$	–4	–2	

$X_4 =$

				δ_i
		140	20	0
120			20	0
	50	50	50	20
0	0	0	20	δ_j

Нули (1, 3), (1, 4), (2, 1) и (2, 4) являются неполными, нуль (3, 2) – полным, выделяем столбцы с 1-го по 3-й.

1-й этап.

Нуль (1, 4) выделяем штрихом, а первую строку – плюсом. Нуль (1,3) выделяем звёздочкой, снимаем выделение с третьего столбца, продолжаем поиск. В третьем столбце нулей нет, а в четвёртом выделяем нуль (2, 4).

Так как невязка в торой сторки – нулевая, выделяем её плюсом. На её пересечении с 1-м столбцом стоит существенный нуль (2, 1), обозначаем позицию звёздочкой и снимаем выделение с первого столбца. Больше в невыделенной части матрицы (серая) нулевых элементов нет. Поиск закончен неудачно.

3-й этап.

Отрицательные элементы выделены однократно, единственный положительный элемент равен 5. Коррекция матрицы с его использованием даёт C_4.

$C_4 =$

	+			
9	*17*	$\dot{0}^{*}$	$\dot{0}^{\nabla}$	+
$\dot{0}^{*}\rightarrow$	*8*	*2*	$\rightarrow\dot{0}^{\nabla}$	+
$\uparrow 0^{\nabla}$	$\ddot{0}$	*–9*	*–7*	

$X_5 =$

				δ_i
		140	20	*0*
100			40	*0*
20	50	50	50	*20*
0	*0*	*0*	*20*	δ_j

1-й этап.

Отмечаем нуль (3, 1) штрихом. Невязка соответствующей строки – положительна, поисковый этап закончился положительно.

2-й этап.

Цепочка соединяет нули с координатами в матрице $C_3\ (3,\ 1)^{\nabla} \Rightarrow (2,\ 1)^{*} \Rightarrow (2,\ 4)^{\nabla}$.

Элемент для её коррекции есть

$$\theta = \min\left\{\delta_3 = 20, \delta_4 = 20, x^{*}_{1,3} = 120, r'_{3,1} = 50, r'_{2,4} = 30\right\} = 20.$$

Получаем план X_5, невязка которого $\Delta = 0$. Следовательно, полученный план – оптимальный.

Целевая функция, соответствующая этому плану

$$L = 140 \times 20 \times 2 + 100 \times 4 + 40 \times 8 + 20 \times 9 + 50 \times (2 + 3 + 6) = 1630.$$

По сравнению с целевой функцией плана перевозок, полученного без учёта ограничений, целевая функция имеет большее значение. Это объясняется тем, что “благодаря” ограничениям не удаётся переместить потребное число единиц товара машрутом с минимальной стоимостью.

2.3.4.8. Решение задачи о назначениях [22, 33, 34]

Задача о *назначениях*, она же задача *распределения* или задача *выбора*, имеет следующую содержательную постановку.

Предположим, что имеется n различных работ: $B_1, B_2, \ldots, B_n$ и столько же исполнителей этих работ (механизмов, например): $A_1, A_2, \ldots, A_n$. Причём, каждый из исполнителей способен выполнять любую работу, но единовременно может быть задействован только на одной из них. Пусть работа механизма A_i при выполнении работы B_j характеризуется некоторой неотрицательной величиной $c_{i,j}$, $i = 1, n, j = 1,n$.

Требуется таким образом распределить работы среди исполнителей, чтобы был достигнут определённый эффект (оптимум), например, достичь максимальной производительности труда или минимального расхода ресурсов.

Формально задача ставится в виде: для заданной матрицы

$$C = \begin{bmatrix} c_{1,1} & c_{1,2} & \dots & c_{1,n} \\ c_{2,1} & c_{2,2} & \dots & c_{2,n} \\ \dots & \dots & \dots & \dots \\ c_{i,1} & c_{i,2} & \dots & c_{1,n} \\ \dots & \dots & \dots & \dots \\ c_{n,1} & c_{2,n} & \dots & c_{n,n} \end{bmatrix}$$

найти распределение (расстановку, выбор и т.д.) $X = [x_{i,j}]$, $i = 1, n, j = 1, n$, обеспечивающие оптимум целевой функции

$$L = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{i,j} \cdot x_{i,j} \to opt . \qquad (2.49)$$

Если матрица C представляет собой ***производительность*** механизмов при выполнении отдельных работ, то задача решается на ***максимум***, а если смысл матрицы C – ***стоимостные или временные издержки***, то задача решается на ***минимум***.

Формально можно полагать, что задачи максимизации и минимизации связаны между собой выражением.

$$c_{i,j}^{Издержки} = 1 \Big/ c_{i,j}^{Производительнось} . \qquad (2.50)$$

Житейски (2.50) можно прокомментировать так, что кто “с огоньком” работает, тот более выгоден, или, если так можно выразиться, лучше окупается.

Попутно заметим, что если, вследствие применения (2.50) числа окажутся дробными, то матрицу C путем умножения на соответствующую степень десятки и отбрасыванием оставшейся дробной части, легко привести к целому виду для удобства расчётов.

Так как при этом (2.49), в силу линейности, возрастёт, то необходимо будет её после расчётов откорректировать делением.

Матрица решения X состоит, в основном, из нулей и содержит всего n единиц, размещающихся в тех позициях, в которые соответствуют назначению i-го исполнителя на j-ю работу. Поэтому, при решении задачи вручную, определение значения (2.49) состоит в суммировании тех элементов исходной матрицы C, которым соответствуют единицы в матрице X.

Методы решения задачи о назначениях

Задача о назначениях представляет собой частный случай транспортной задачи без ограничения на пропускную способность коммуникаций.

Частности состоят в следующем:

- число поставщиков равно числу потребителей, отчего матрица C – квадратная;
- так как имеется условие о выполнении одним исполнителем в текущий момент времени только одной работы, то вектора с объёмами производства и объёмами потребления следует сделать единичными.

Исходя из этих допущений, для решения задачи о назначениях пригоден любой метод решения ТЗ, учитывающий матрицу стоимостей.

Понятно, что появились методы, учитывающие особенности задачи о назначениях. Наиболее популярна из них – модификация венгерского метода.

Ниже нами будет рассмотрен, для разнообразия, алгоритм решения задачи ***максимизации***. В этом случае полагается, что элементы матрицы C есть производительность i-го исполнителя при постановке его на выполнение j-й работы. Метод использует понятие “независимого нуля”.

Независимым нулём называется нуль матрицы C, не содержащий в строке и в столбце, на пересечении которых он находится, других независимых нулей.

Алгоритм решения задачи о назначениях с максимизацией целевой функции

Алгоритм имеет классическую структуру, характерную для всех венгерских методов.

Состоит из предварительного этапа и трёхэтапной итерации, в начале которой выполняется разметка.

Предварительный этап.

Состоит в построении матрицы C_0 и расстановке независимых нулей.

- В каждом столбце матрицы C отыскивается максимальный элемент, из которого затем вычитаются все элементы этого столбца. Результат записывается на место вычитаемого. В результате, матрица C преобразуется в матрицу $C`$.
- В каждой строке матрицы $C`$ отыскивается минимальный элемент, который затем вычитается из всех элементов этой строки. В результате, матрица $C`$ преобразуется в матрицу C_0.

- Произвольно отмечаем нуль в первом столбце матрицы C_0 звёздочкой, полагая его независимым. Просматриваем по порядку остальные столбцы матрицы, руководствуясь определением. Если в рассматриваемых столбцах есть нулевой элемент, стоящий на строке, где нет нуля со звёздочкой, то его полагаем независимым, отмечаем звёздочкой и переходим к рассмотрению следующего столбца.

Заметим, что если бы задача решалась на минимум, то, в этом случае, построение матрицы C_0 не отличалось бы от обычного венгерского метода.

Проверка условия окончания

Подсчитываем число нулей со звёздочками (независимых нулей). Если их число равно размерности матриц n, то достигнут оптимум.

Необходимо рассчитать целевую функцию (2.49). План X, содержащий оптимальную расстановку, полностью определяется текущей матрицей C: нули, отмеченные звёздочками, соответствуют, единичным позициям оптимального плана, все прочие позиции – нулевые.

ИТЕРАЦИОННАЯ ЧАСТЬ АЛГОРИТМА

По завершении итерации, число независимых нулей в текущей матрице C увеличивается на один.

Таким образом, точное число итераций есть

$$N_{\text{итераций}} = n - n^*, \tag{2.51}$$

где n – размерность матрицы, n^* - число независимых нулей.

Возможное число итераций, даже в начале счёта не более $n - 2$.

Разметка.

Разметка текущей матрицы C выполняется в начале итерации и сохраняется до её конца с теми изменениями, которые вносятся в неё по мере выполнения алгоритма.

Необходимо выделить знаком «плюс» j-е столбцы матрицы C, в которых присутствуют независимые нули.

Этап 1 – этап поиска.

Область поиска: невыделенная часть матрицы C – невыделенные столбцы и строки.

Цель поиска: найти в невыделенной части матрицы C нуль, стоящий в строке, в которой нет независимого нуля.

Поисковый этап заканчивается, как и в любом алгоритме венгерского метода, одним из случаев:

- если все нули матрицы C находятся в выделенной части, то необходимо перейти к этапу 3 – эквивалентных преобразований матрицы C;

- поиск завершился успешно, найден ноль в строке, где нет других независимых нулей. В этом случае далее выполняется этап 2 – построение цепочки и коррекция плана назначений.

В ходе поиска невыделенная часть матрицы *C* просматривается по столбцам сверху вниз а столбцы – слева направо.

Пусть среди элементов найден ноль. Его отмечают апострофом (штрихом) и анализируют строку, на которой он находится.

Если нет других независимых нулей (со звёздами), то этот ноль со штрихом является искомым, а поиск заканчивается успешно.

Если в строке уже имеется независимый нуль, то текущая стока выделяется знаком «плюс», и просматривается по местам её пересечения с выделенными столбцами. Если в месте пересечения стоит нуль со звёздочкой (*), то знак выделения над столбцом уничтожают, обводя кружком или заключая в скобки. Столбец становится невыделенным и делается доступным для поиска. Поиск далее продолжают по этому столбцу со снятым выделением.

Этап 2 – этап построения цепочки и коррекции плана назначений.

1. Цепочка ***не замкнута***, составляется из нулей со штрихом (0’) и нулей со звёздочками (0^*), содержит нечётное число элементов, и, в принципе, может состоять и из одного нуля со штрихом.

2. Цепочка начинается от последнего найденного нуля со штрихом 0’ к нулю со звездой 0^* по столбцу ($0' \to 0^*$), далее, по направлению под 90° к предыдущему, по строке от нуля со звездой к нулю со штрихом ($0^* \to 0'$) и так далее. На нечётных местах цепочки будут стоять нули со штрихом, а на чётных – нули со звёздами. Цепочка начинается в строке, в которой нет независимых нулей, и заканчивается в столбце, который в ходе разметки избежал выделения.

При переписи матрицы *C* апострофы (штрихи) возле нулей, составляющих цепочку, заменяются звёздами, а звёздочку – устраняют. В результате число независимых нулей увеличивается на единицу.

Этап 3 – этап эквивалентного преобразования матрицы C.

Этот этап ничем не отличается от изложенного ранее, приводится здесь для полноты восприятия.

1. Среди невыделенных элементов текущей матрицы *C* выбирается минимальный положительный элемент $h > 0$. Этот элемент называется ***корректирующим***.

2. Корректирующий элемент вычитается от невыделенных строк матрицы *C*.

3. Корректирующий элемент прибавляется к выделенным столбцам матрицы *C*.

Пример. Решить задачу о назначениях на максимум.

$C =$

9	7	8	6	3	9
2	8	6	8	5	4
4	2	7	7	7	5
3	5	5	2	6	4
6	4	4	5	6	8
5	6	6	3	4	7

Предварительный этап.

Максимальные элементы в столбцах C выделены серым. В результате преобразования исходной матрицы по столбцам имеем

$C' =$

0	1	0	2	4	0
7	0	2	0	2	5
5	6	1	1	0	4
6	3	5	6	1	5
3	4	4	3	1	1
4	2	2	5	3	2

Минимальные элементы в строках C' выделены серым. После вычитания получаем C_0 вида

$C_0 =$

(+)	(+)	(+)		+	+	
0^{*}	1	0^{∇}	2	4	0	+
7	0^{*}	2	0^{∇}	2	5	+
5	6	1	1	0^{*}	4	
5	2	4	5	0	4	
2	3	3	2	0	0^{*}	
2	0^{∇}	0^{*}	3	1	0	+

Расставляем звёзды при независимых нулях: (1, 1),. (2, 2), (6, 3), (3, 5) и (5, 6).

Проверка условия окончания.

Число независимых нулей равно пяти, следовательно, задача решится за одну итерацию.

Итерация.

Разметка.

Размечаем 1, 2, 3, 5, 6 столбцы плюсами.

1-й этап, поиск.

Отмечаем нуль (2, 4) штрихом. На этой строке – независимый ноль, выделяем её, а выделение со второго столбца снимаем.

Во втором столбце отмечаем нуль (6, 2) штрихом, и, так как в шестой строке стоит независимый нуль, выделяем и её. После чего, снимаем выделение с третьего столбца, и отмечаем нуль (1, 3).

Первая строка выделяется плюсом, а выделение с первого столбца снимается. Поиск завершился неудачно.

3-й этап.

В невыделенной части матрицы (серый фон) находим корректирующий элемент. Он равен единице (таких элементов в матрице целых два). Корректируем матрицу C_0, получаем C_1.

					(+)	+	
	0^*	1	0^∇	2	5	1	+
	7	$0^*\rightarrow$	2	$\rightarrow 0^\nabla$	3	6	+
$C_1=$	4	5	$\downarrow 0^\nabla$	0	$\leftarrow 0^*$	4	+
	4	1	1	4	$\uparrow 0^\nabla$	4	
	1	2	2	1	0	0^*	
	2	$\uparrow 0^\nabla$	$\leftarrow 0^*$	3	2	1	+

1-й этап, снова поиск.

Нуль (3, 3) отмечаем штрихом, третья строка отмечается плюсом, и снимается выделение с пятого столбца.

В четвёртой строке отмечаем нуль (4, 5), который оказывается искомым. Этап удачно завершён.

2-й этап, коррекция плана.

Строим цепочку (4, 5) → (3, 5) → (3, 3) → (6, 3) → (6, 2) → (2, 2) → (2, 4). Проводимые согласно индексации замены даю следующую топологию размещения независимых нулей.

	0^*	1	0	2	5	1
	7	0	2	0^*	3	6
$C_1=$	4	5	0^*	0	0	4
	4	1	1	4	0^*	4
	1	2	2	1	0	0^*
	2	0^*	0	3	2	1

Число независимых нулей равно шести, расчёты закончены. Имеем следующий план расстановки исполнителей по работам:

$$X = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Целевая функция полученного плана $L = 9 + 8 + 7 + 6 + 8 + 6 = 44$.

Решение предписывает, что 1-й исполнитель задействован на выполнении 1-й работы, 2-й – на 4-й, 3-й – на 3-й, 4-й – на 5-й, 5-й – на 6-й, а 6-й – на 2-й.

2.3.4.9. Вопросы для самоконтроля

1. Каковы недостатки алгоритма метода северо-западного угла?
2. В чём привлекательность метода минимальной стоимости?
3. Почему метод штрафов лучше метода минимальной стоимости?
4. Какой опорный план называется вырожденным и почему?
5. Как связаны между собой ЗЛП и ТЗ?
6. Что такое потенциал, и в чём его практический смысл?
7. Какова геометрическая интерпретация опорного плана *Т*-задачи?
8. В чём состоит понятие баланса в *Т*-задачах?
9. Для чего нужны цепочки при коррекции планов?
10. В чём заключается физический смысл построения цепочки?
11. Почему цепочки метода потенциалов замкнутые?
12. Каковы приёмы, применяемые для поддержания невырожденности плана?
13. В чём заключаются трудности, связанные с реализацией метода потенциалов на ЭВМ?
14. Какая транспортная модель называется замкнутой?
15. Для чего проверяется условие баланса и выполняется балансировка?
16. Что такое невязка?
17. Почему для венгерского метода нет необходимости в использовании опорного плана?
18. Что такое существенные и несущественные нули?
19. Как прогнозируется число итераций, оставшихся до получения оптимального решения?

20. Что является целью этапа поиска венгерского метода?

21. Где выполняется поиск?

22. В чём состоит назначение и сущность этапа эквивалентных преобразований матрицы *C*?

23. В чем сходство и различие при построении начальных планов для решения задачи венгерским методом и методом потенциалов?

24. Каковы, на Ваш взгляд, преимущества и недостатки обоих методов решения ТЗ? Обоснуйте высказываемые суждения.

25. Почему венгерский метод удобен при реализации на ЭВМ?

26. Поясните физический смысл, заключенный в процедуре построения цепочки.

27. Почему цепочка, которая строится в алгоритме метода потенциалов, замкнута, а в венгерском методе нет?

28. Каковы признаки неразрешимости *Td*-задачи?

29. Обоснуйте, почему *Td*-задачу трудно или невозможно решить методом потенциалов?

30. С чем связано увеличение целевой функции *Td*-задачи по сравнению с обычной *T*-задачей?

31. В чём состоят особенности применения алгоритма венгерского метода при наличии ограничений?

32. Приведите постановки *T*-задач применительно к технике передачи информации и компьютерным сетям.

34. Почему *Td*-задача не всегда может быть решена?

35. Как задается условие дискретности в задаче о назначениях?

36. Как связаны между собой задача о назначениях и Т-задача?

37. Всегда ли разрешима задача данного типа? Ответ обосновать.

38. Какой ноль называется независимым?

39. Какой алгоритм из известных алгоритмов решения транспортных задач кажется Вам наиболее эффективным? Дать обоснование.

40. Что будет, если наложить условие на допуск исполнителей к определенным работам?

41. Что, на Ваш взгляд, поменяется в постановке и алгоритме задачи, если каждый исполнитель будет в состоянии выполнять более одной работы, а каждая работа – быть исполнимой более чем одним исполнителем?

42. Как оценить число итераций, оставшихся до получения оптимального решения?