А.Н.Сергеев mail: a_cepreeb@mail.ru

Решение задачи о назначении венгерским методом

Некоторая туристическая фирма планирует экскурсии СПб-Хельсинки. Принимается следующее расписание туристических автобусов:

Хельсинки	→	СПб	СПб	→	Хельсинки
	а			1	
8.00	→	14.00	6.00	→	12.00
	б			2	
16.00	→	22.00	8.00	→	14.00
	в			3	
24.00	→	6.00	24.00	→	6.00

Обслуживающие эти рейсы бригады могут проживать как в СПб так и в Хельсинки. Время ожидания бригадой обратного рейса не должно превышать 24 часа и должно быть не менее 4-х часов. Задача менеджера туристической фирмы в том, чтобы минимизировать общее время нахождения бригад вне дома. Так как, оплата бригады не зависит от того, находятся они в пути или ожидают своего возвращения домой.

Составим таблицы потерянного времени для случаев, когда все бригады живут в Хельсинки и когда все бригады живут в СПб. Пусть, например, бригада живёт в Хельсинки и назначается на рейс а1. Тогда отправления она будет ожидать (24 – 14)+6=16 часов. Таким образом имеем:

Бригады живут в Хельсинки

	а	б	в
1	16	8	0
2	18	10	2
3	10	2	18

Бригады живут в СПб

	а	б	в
1	20	4	12
2	18	2	10
3	2	10	18

Составим теперь сводную таблицу, куда войдут наименьшие времена ожиданий, но большие 4-х

	а	б	в
1	16	8	12
2	18	10	10
3	10	10	18

Теперь менеджеру надо каждой строке сводной таблицы поставить в соответствие ровно один столбец. Тогда выбор бригад и назначений будет сделан. Например: 1а, 2б, 3в и суммарное время ожидания – 44 часа. Или 1в, 2б, 3а и тогда суммарное время ожидания – 32 часа.

Выбор для каждой строки единственного столбца матрицы затрат – это назначение. Задача о назначении заключается в том, чтобы сделать назначение с минимальными затратами.

Перейдём к рассмотрению алгоритма решения этой задачи.

Рассмотрим вначале задачу о назначении без издержек. Пусть, например, имеется m рабочих и n работ. Составим матрицу M, состоящую из 0 и 1, таким образом, что $M_{i,j} = 1$, если $i-$й рабочий может выполнять $j-$ю работу и $M_{i,j} = 0$ в противном случае.

Максимальным будет такое назначение, при котором будет выполняться максимальное число работ.

Например:

1		1			1
		1			
		1		1	
		1	1		

Алгоритм:

1. Выбрать строку, состоящую из минимального числа единиц, или одну из таких строк. Пометить одну из единиц в этой строке.

2. Прочеркнуть строку и столбец, пересекающиеся по отмеченной единице.
3. Повторять операции 1,2 пока все строки или все столбцы будут прочёркнуты.

Рассмотрим теперь задачу о назначении, связанную с затратами. Пусть каждый из m рабочих может выполнять любую из m работ. Но назначение $i-$го рабочего на $j-$ю работу требует $M_{i,j}$ затрат.

Например.

3	5	7	2	1
4	6	7	3	1
2	1	3	4	5
6	3	2	7	8
5	4	3	1	9

Требуется найти такое назначение, чтобы суммарные затраты на выполнение работ были минимальны.

Затраты по строкам на назначение вычисляются по формуле

$C_{\beta_1,\beta_2,\ldots,\beta_m} = M_{1,\beta_1} + M_{2,\beta_2} + \ldots + M_{m,\beta_m}$, где $\beta_1, \beta_2, \ldots, \beta_m$ все возможные перестановки столбцов $1,2,\ldots,m$.

Затраты по столбцам на назначение вычисляются по формуле

$C^{\alpha_1,\alpha_2,\ldots,\alpha_m} = M_{\alpha_1,1} + M_{\alpha_2,2} + \ldots + M_{\alpha_m,m}$, где $\alpha_1, \alpha_2, \ldots, \alpha_m$ всевозможные перестановки строк $1,2,\ldots,m$.

Лемма 1. Если вычесть из каждого значения затрат $i-$й строки произвольную постоянную u_i, то оптимальное назначение не изменится.

Доказательство. Пусть назначение $\beta_1, \beta_2, \ldots, \beta_m$ оптимальное. То есть,

$$C_{\beta_1,\beta_2,\ldots,\beta_m} \le C_{\beta'_1,\beta'_2,\ldots,\beta'_m}.$$

Вычтем, например, из первой строки величину u_1. Тогда стоимость назначения $\beta_1, \beta_2, \ldots, \beta_m$ будет вычисляться по формуле
$(M_{1,\beta_1} - u_1) + M_{2,\beta_2} + \ldots + M_{m,\beta_m} = C_{\beta_1,\beta_2,\ldots,\beta_m} - u_1$.

Аналогично, стоимость назначения $\beta'_1, \beta'_2, \ldots, \beta'_m$:

$$(M_{1,\beta'_1} - u_1) + M_{2,\beta'_2} + \ldots + M_{m,\beta'_m} = C_{\beta'_1,\beta'_2,\ldots,\beta'_m} - u_1.$$

Очевидно, что $C_{\beta_1,\beta_2,\ldots,\beta_m} - u_1 \le C_{\beta'_1,\beta'_2,\ldots,\beta'_m} - u_1$. **Теорема доказана.**

Аналогично доказывается

Лемма 2. Если вычесть из каждого значения затрат j−го столбца произвольную постоянную v_j, то оптимальное назначение не изменится.

Найдём такие u_i, $i \in 1:m$, v_j, $j \in 1:m$, что $M_{i,j} \ge u_i + v_j$.
Это можно сделать с помощью операции приведения матрицы затрат $\{M_{i,j}\}$.

Алгоритм:

1. В каждой строке находим минимальный элемент и вычитаем его из строки. Получаем матрицу затрат приведённую по строкам.
2. В каждом столбце находим минимальный элемент и вычитаем его из строки. Получаем приведённую матрицу.

3	5	7	2	1
4	6	7	3	1
2	1	3	4	5
6	3	2	7	8
5	4	3	1	9

2	4	6	1	0
3	5	6	2	0
1	0	2	3	4
4	1	0	5	6
4	3	2	0	8

1	4	6	1	0
2	5	6	2	0
0	0	2	3	4
3	1	0	5	6
3	3	2	0	8

Для приведённой матрицы стоимости затраты на назначения можно вычислить по формуле

$$\hat{C}_{\beta_1,\beta_2,\ldots,\beta_m} = C_{\beta_1,\beta_2,\ldots,\beta_m} - \sum_i u_i - \sum_j v_j .$$

Обозначим $S = \sum_i u_i + \sum_j v_j$.

Очевидно, что если найдено такое назначение $\beta_1, \beta_2, \ldots, \beta_m$, и такие u_i, $i \in 1:m$, v_j, $j \in 1:m$, что $C_{\beta_1,\beta_2,\ldots,\beta_m} = \sum_i u_i + \sum_j v_j$, то затраты на назначения будут минимальны, а величина S максимальна.

При этом, для приведённой матрицы по значениям u_i, $i \in 1:m$, v_j, $j \in 1:m$

$$\hat{C}_{\beta_1,\beta_2,\ldots,\beta_m} = C_{\beta_1,\beta_2,\ldots,\beta_m} - \sum_i u_i - \sum_j v_j = 0 .$$

Последнее равенство является критерием оптимального назначения. То есть, если для приведённой матрицы можно сделать назначение с нулевой суммой затрат, то это назначение оптимально. Назначения для приведённой матрицы можно выполнять по алгоритму задачи о назначении без затрат.

Алгоритм:

1. Выбрать строку, состоящую из минимального числа нулей, или одну из таких строк. Пометить одну из нулей в этой строке.
2. Прочеркнуть строку и столбец, пересекающиеся по отмеченному нулю.
3. Повторять операции 1,2 пока все строки или все столбцы будут прочёркнуты.

Таким образом, для задачи о назначении ставится двойственная задача ЛП

$$\sum_i u_i + \sum_j v_j \to \max$$

$$u_i + v_j \le M_{i,j} \ , \ i \in 1:m, \ v_j, j \in 1:m$$

$u_i \geq 0,\ v_j \geq 0$.

Приведение матрицы – это нахождение некоторого начального плана двойственной задачи. Будем этот план улучшать.

1. Прочеркнём минимальное число рядов приведённой матрицы, чтобы зачеркнуть все нули.

1	4	6	1	0
2	5	6	2	0
0	0	2	3	4
3	1	0	5	6
3	3	2	0	8

2. В оставшейся матрице находим наименьшее число $d = 1$. Это число можно а) вычитать от не зачёркнутых строк и прибавлять к зачёркнутым столбцам, б) вычитать от не зачёркнутых столбцов и прибавлять к зачёркнутым строкам. При этом приведённая матрица останется неотрицательной.

Обозначим зачёркнутые строки индексным множеством M_r, а зачёркнутые столбцы M_c.

А) Получим новые двойственные векторы, вычитая число d от не зачёркнутых строк и прибавляя к зачёркнутым столбцам:

$$\hat{u}_i = u_i - d\ ,\ i \in M \setminus M_r, \qquad \hat{v}_j = v_j, \qquad j \in M \setminus M_c$$

$$\hat{u}_i = u_i, \qquad i \in M, \qquad \hat{v}_j = v_j + d, \quad j \in M$$

Тогда новая величина

$$\hat{S} = \sum_{i \in M \setminus M_r} (u_i - d) + \sum_{i \in M} u_i + \sum_{j \in M} v_j + \sum_{j \in M \setminus M_c} (v_j + d) =$$

$$= \sum_{i \in M \setminus M_r} u_i + \sum_{i \in M} u_i + \sum_{j \in M} v_j + \sum_{j \in M \setminus M_c} v_j - \sum_{i \in M \setminus M_r} d + \sum_{j \in M \setminus M_c} d =$$

$$= S - \sum_{i \in M \setminus M_r} d + \sum_{j \in M \setminus M_c} d .$$

Замечаем, что $S' \leq S$, если вычитать от рядов, большее количество которых не зачёркнуто.

В нашем примере вычитать нужно от не зачёркнутых строк и прибавлять к зачёркнутым столбцам.

1	4	6	1	0
2	5	6	2	0
0	0	2	3	4
3	1	0	5	6
3	3	2	0	8

0	3	6	1	0
1	4	6	2	0
0	0	3	4	5
2	0	0	5	6
1	1	2	0	8

Теперь можно сделать оптимальное назначение.

Алгоритм Венгерского метода.

1. Приводим матрицу – находим начальное приближение двойственного вектора.
2. Пытаемся сделать назначение с нулевой стоимостью.
3. Прочеркнём минимальное число рядов приведённой матрицы, чтобы зачеркнуть все нули.
4. В оставшейся подматрице находим наименьшее число d.
5. ***if*** не зачёркнутых строк больше, чем не зачёркнутых столбцов

then вычитаем число d от не зачёркнутых строк и прибавляем к зачёркнутым столбцам

else вычитаем число d от не зачёркнутых столбцов и прибавляем к зачёркнутым строкам.

Задания.

4	3	5	8	6	4
5	5	6	8	5	4
4	2	5	7	7	5
5	7	3	4	3	6
4	7	4	5	4	1
6	3	8	6	5	2

2	4	5	1	4	1
5	4	3	2	2	4
2	2	2	5	6	2
3	4	4	1	4	3
4	5	3	2	4	3
5	4	5	3	6	4

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

4	3	6	10	1	5
5	5	7	10	1	1
4	2	6	9	1	6
5	7	4	6	1	7
3	3	4	5	8	5
6	3	9	8	1	3

3	6	4	6	4	2
1	3	2	1	3	5
5	5	5	5	1	4
4	5	6	7	1	9
2	6	6	6	1	7
5	6	4	5	1	5

3	2	7	8	5	5
3	5	7	6	6	10
5	2	3	2	1	3
4	2	6	9	2	7
2	8	9	8	2	3
1	9	8	8	3	9

4	2	6	9	1	6
3	3	4	5	8	5
4	3	6	10	1	5
5	7	4	6	1	7
6	3	9	8	1	3
5	5	7	10	1	1

6	6	2	4	3	4
1	3	5	3	1	2
5	5	4	1	5	5
7	5	9	1	4	6
6	6	7	1	2	6
5	6	5	1	5	4

2	8	9	8	2	3
1	9	8	8	3	9
3	5	7	6	6	10
4	2	6	9	2	7
5	2	3	2	1	3
3	2	7	8	5	5

5	7	3	4	3	6
4	7	4	5	4	1
4	3	5	8	6	4
6	3	8	6	5	2
4	2	5	7	7	5
5	5	6	8	5	4

5	4	3	2	2	4
3	4	4	1	4	3
4	5	3	2	4	3
2	4	5	1	4	1
5	4	5	3	6	4
2	2	2	5	6	2

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

[illegible]	[illegible]	4	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

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

3	6	6	3	5	5
3	6	7	2	7	4
3	7	6	3	7	4
3	6	8	2	7	2
6	2	3	1	9	5
5	4	5	6	3	1

4	3	5	8	6	4
5	5	6	8	5	4
4	2	5	7	7	5
5	7	3	4	3	6
4	7	4	5	4	1
6	3	8	6	5	2

2	4	5	1	4	1
5	4	3	2	2	4
2	2	2	5	6	2
3	4	4	1	4	3
4	5	3	2	4	3
5	4	5	3	6	4

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

4	3	6	10	1	5
5	5	7	10	1	1
4	2	6	9	1	6
5	7	4	6	1	7
3	3	4	5	8	5
6	3	9	8	1	3

3	6	4	6	4	2
1	3	2	1	3	5
5	5	5	5	1	4
4	5	6	7	1	9
2	6	6	6	1	7
5	6	4	5	1	5

3	2	7	8	5	5
3	5	7	6	6	10
5	2	3	2	1	3
4	2	6	9	2	7
2	8	9	8	2	3
1	9	8	8	3	9

4	2	6	9	1	6
3	3	4	5	8	5
4	3	6	10	1	5
5	7	4	6	1	7
6	3	9	8	1	3
5	5	7	10	1	1

6	6	2	4	3	4
1	3	5	3	1	2
5	5	4	1	5	5
7	5	9	1	4	6
6	6	7	1	2	6
5	6	5	1	5	4

2	8	9	8	2	3
1	9	8	8	3	9
3	5	7	6	6	10
4	2	6	9	2	7
5	2	3	2	1	3
3	2	7	8	5	5

5	7	3	4	3	6
4	7	4	5	4	1
4	3	5	8	6	4
6	3	8	6	5	2
4	2	5	7	7	5
5	5	6	8	5	4

5	4	3	2	2	4
3	4	4	1	4	3
4	5	3	2	4	3
2	4	5	1	4	1
5	4	5	3	6	4
2	2	2	5	6	2

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

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

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

3	6	6	3	5	5
3	6	7	2	7	4
3	7	6	3	7	4
3	6	8	2	7	2
6	2	3	1	9	5
5	4	5	6	3	1

5	6	5	5	2	5
1	4	4	6	1	5
8	9	5	5	3	3