Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика, искусственный интеллект и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

ОТЧЕТ

по лабораторной работе № 1

по курсу «Методы вычислений»

Вариант № 9

Студент Сапожков А. М./ИУ7-13М — (Группа)

(Подпись, дата) — Власов П. А. (И. О. Фамилия)

Преподаватель

(Подпись, дата)

OcFFL

OcFFL (И. О. Фамилия)

2024 г.

1 Теоретическая часть

Цель работы: изучение венгерского метода решения задачи о назначениях.

Задание:

1. Реализовать венгерский метод решения задачи о назначениях в виде программы на ЭВМ.

2. Провести решение задачи с матрицей стоимостей, заданной в индивидуальном варианте, рассмотрев два случая:

   - задача о назначениях является задачей минимизации,
   - задача о назначениях является задачей максимизации.

1.1 Содержательная и математическая постановки задачи о назначениях

Содержательная постановка: имеется n работ и n исполнителей; стоимость выполнения i-ой работы j-ым исполнителем составляет $c_{ij} \geqslant 0$ единиц. Требуется распределить все работы между исполнителями так, чтобы

- каждый исполнитель выполнял 1 работу;
- общая стоимость выполнения всех работ была min.

Введём управляемые переменные:

$$x_{ij} = \begin{cases} 1, \text{ если } i\text{-ую работу выполняет } j\text{-ый работник,} \\ 0, \text{ иначе;} \end{cases}$$
$$i, j = \overline{1; n}. \tag{1.1}$$

Из переменных x_{ij}, $i, j = \overline{1; n}$, составим

$$X = (x_{ij})_{i,j=\overline{1;n}}, \tag{1.2}$$

которую назовём матрицей назначений.

Стоимости выполнения работ также записываем в матрицу

$$C = (c_{ij})_{i,j=\overline{1;n}}, \tag{1.3}$$

называемой матрицей стоимостей.

Тогда:

1. Стоимость выполнения работ:

$$f = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}. \tag{1.4}$$

2. Условие того, что i-ую работу выполнит ровно один работник:

$$\sum_{j=1}^{n} x_{ij} = 1,\ i = \overline{1;n}. \tag{1.5}$$

3. Условие того, что j-ый работник выполнит ровно одну работу:

$$\sum_{i=1}^{n} x_{ij} = 1,\ j = \overline{1;n}. \tag{1.6}$$

Таким образом приходим к **математической постановке**:

$$\begin{cases} f = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \to min, \\ \sum_{j=1}^{n} x_{ij} = 1,\ i = \overline{1;n}, \\ \sum_{i=1}^{n} x_{ij} = 1,\ j = \overline{1;n}, \\ x_{ij} \in \{0,1\},\ i,j = \overline{1;n}. \end{cases} \tag{1.7}$$

1.2 Исходные данные варианта №9

$$C = \begin{bmatrix} 4 & 7 & 1 & 5 & 5 \\ 6 & 8 & 3 & 7 & 6 \\ 6 & 4 & 5 & 7 & 7 \\ 4 & 2 & 3 & 4 & 9 \\ 8 & 1 & 8 & 3 & 8 \end{bmatrix} \tag{1.8}$$

1.3 Краткое описание венгерского метода

Схема венгерского метода решения задачи о назначениях представлена на рисунках 1.1-1.3.

Рисунок 1.1 – Схема венгерского метода решения задачи о назначениях, часть 1

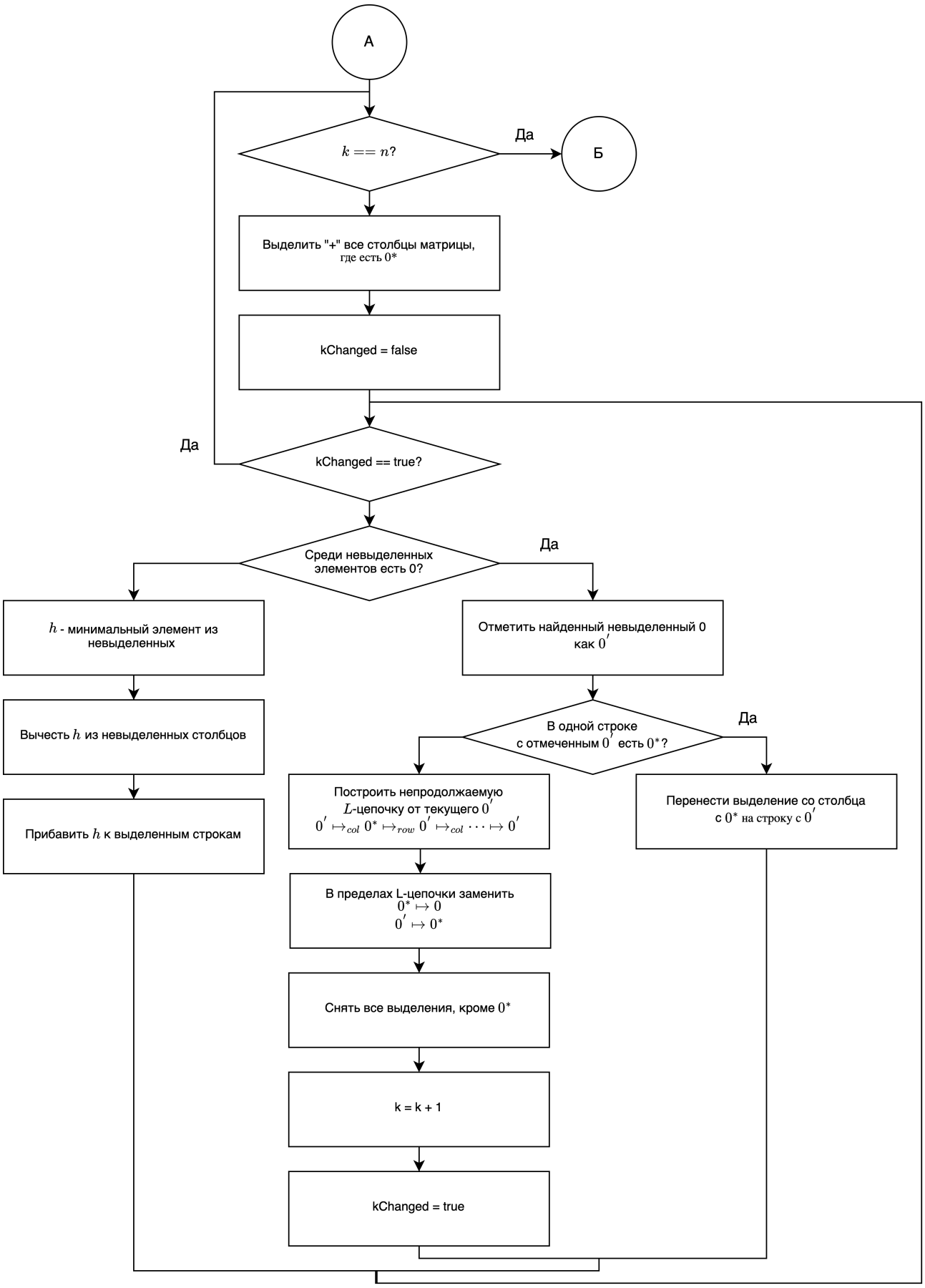

Рисунок 1.2 – Схема венгерского метода решения задачи о назначениях, часть 2

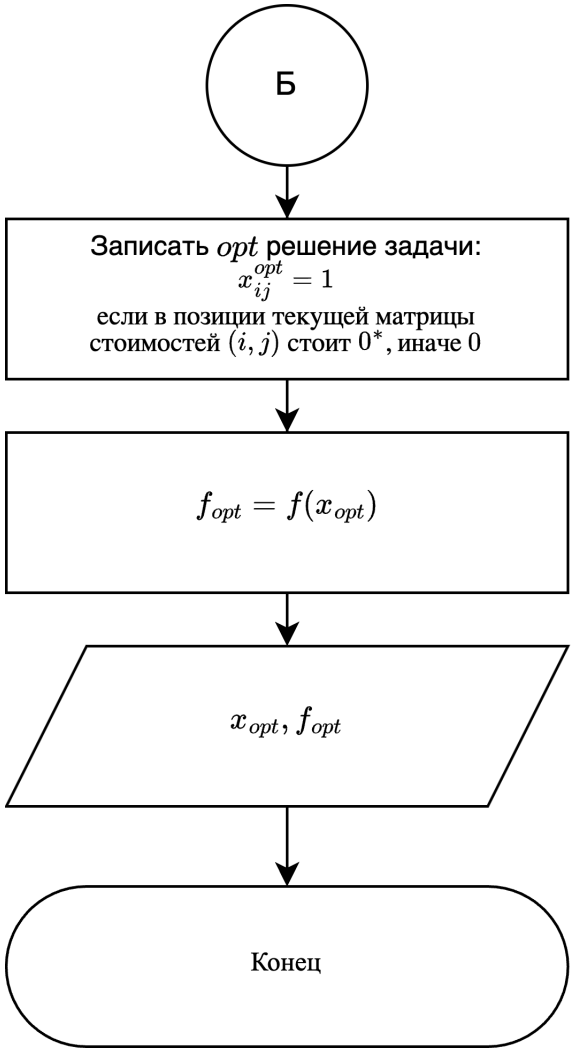

Рисунок 1.3 – Схема венгерского метода решения задачи о назначениях, часть 1

2 Практическая часть

2.1 Текст программы

Листинг 2.1 – Исходный код программы.

```
function hungarian_method()
    clc;
    findMax = false;
    debugMode = true;

    matr = [
        4 7 1 5 5;
        6 8 3 7 6;
        6 4 5 7 7;
        4 2 3 4 9;
        8 1 8 3 8
    ];

    disp('Матрица стоимостей (9 вариант):');
    disp(matr);

    C = matr;

    if findMax == true
        C = convertToMinimizationProblem(matr);

        if debugMode == true
            disp('Матрица стоимостей после сведения к задаче
               минимизации:');
            disp(C);
        end
    end

    C = updateColumns(C);
    if debugMode == true
        disp("Результат вычитания наименьшего элемента по
           столбцам:");
        disp(C);
```

```
end

C = updateRows(C);
if debugMode == true
    fprintf("Результат вычитания наименьшего элемента по
       строкам:\n");
    disp(C);
end

[numRows, numCols] = size(C);

matrSIZ = makeSIZ(C);
if debugMode == true
    fprintf('Начальная СНН:\n');
    printSIZ(C, matrSIZ);
end

k = sum(sum(matrSIZ));
if debugMode == true
    fprintf('\nЧисло нулей в построенной СНН: k = %d\n', k);
end

iteration = 1;
while k < numCols
    if debugMode == true
        fprintf('\nk < n = %d ==> СНН нужно улучшать\n',
           numCols);
        fprintf('_____________________________ Итерация %d
           _____________________________\n', iteration);
    end

    matrStreak = zeros(numRows, numCols);
    selectedColumns = sum(matrSIZ);
    selectedRows = zeros(numRows);
    selection = makeSelection(numRows, numCols,
       selectedColumns);

    if debugMode == true
```

```
    fprintf('\nРезультат выделения столбцов, в которых
       стоит 0*:\n');
    printMarkedMatr(C, matrSIZ, matrStreak,
       selectedColumns, selectedRows);
end

kChanged = false;
streakPoint = [-1 -1];
while kChanged == false
    streakPoint = findStreak(C, selection);
    if streakPoint(1) == -1
        if debugMode == true
            fprintf("\nСреди невыделенных элементов нет 0,
               преобразуем матрицу:\n");
        end

        C = updateMatrNoZero(C, numRows, numCols,
           selection, selectedRows, selectedColumns);

        if debugMode == true
            printMarkedMatr(C, matrSIZ, matrStreak,
               selectedColumns, selectedRows);
        end

        streakPoint = findStreak(C, selection);
    end

    matrStreak(streakPoint(1), streakPoint(2)) = 1;
    if debugMode == true
        fprintf("\nСреди невыделенных элементов есть 0,
           отметим 0':\n");
        printMarkedMatr(C, matrSIZ, matrStreak,
           selectedColumns, selectedRows);
    end

    zeroStarInRow = getZeroStarInRow(streakPoint, numCols,
       matrSIZ);
    if zeroStarInRow(1) == -1
```

```
            [matrStreak, matrSIZ] = makeLChain(numRows,
               numCols, streakPoint, matrStreak, matrSIZ,
               debugMode);
            kChanged = true;
        else
            % снять выделение со столбца с 0*
            selection(:, zeroStarInRow(2)) = selection(:,
               zeroStarInRow(2)) - 1;
            selectedColumns(zeroStarInRow(2)) = 0;

            % перенести выделение на строку с 0'
            selection(zeroStarInRow(1), :) =
               selection(zeroStarInRow(1), :) + 1;
            selectedRows(zeroStarInRow(1)) = 1;
            if debugMode == true
                fprintf("\nВ одной строке с 0' есть 0*,
                   перебросим выделение со столбца на
                   строку:\n");
                printMarkedMatr(C, matrSIZ, matrStreak,
                   selectedColumns, selectedRows);
            end
        end
    end

    k = sum(sum(matrSIZ));
    if debugMode == true
        fprintf("\nВ пределах L-цепочки 0* заменем на 0, а 0'
           на 0*:\n");
        printSIZ(C, matrSIZ);
        fprintf('\nЧисло нулей в построенной СНН: k = %d\n',
           k);
    end

    iteration = iteration + 1;
end

if debugMode == true
    fprintf("\nКонечная СНН:\n");
    printSIZ(C, matrSIZ);
```

```
    end

    fprintf("\nX: \n");
    disp(matrSIZ);

    fOpt = getFOpt(matr, matrSIZ);
    fprintf("f_opt = %d", fOpt);
end

% Найти первый нулевой элемент среди невыделенных, в одной строке
   с которым не стоит 0*
function [streakPoint] = findStreak(matr, selection)
    streakPoint = [-1 -1];
    [numRows, numCols] = size(matr);
    for i = 1 : numCols
        for j = 1 : numRows
           if selection(j, i) == 0 && matr(j, i) == 0
                streakPoint(1) = j;
                streakPoint(2) = i;
                return;
           end
        end
    end
end

function [] = printSIZ(matr, matrSIZ)
    [numRows, numCols] = size(matr);

    for i = 1 : numRows
        for j = 1 : numCols
            if matrSIZ(i, j) == 1
                fprintf("\t%d*", matr(i, j));
            else
                fprintf("\t%d", matr(i, j));
            end
        end
        fprintf("\n");
    end
end
```

```
function [] = printMarkedMatr(matr, matrSIZ, matrStreak,
   selectedCols, selectedRows)
    [numRows, numCols] = size(matr);

    for i = 1 : numRows
        if selectedRows(i) == 1
            fprintf("+");
        end

        for j = 1 : numCols
            if matrSIZ(i, j) == 1
                fprintf("\t%d*\t", matr(i, j));
            elseif matrStreak(i, j) == 1
                fprintf("\t%d'\t", matr(i, j));
            else
                fprintf("\t%d\t", matr(i, j));
            end
        end
        fprintf("\n");
    end

    for i = 1 : numCols
        if selectedCols(i) == 1
            fprintf("\t+\t");
        else
            fprintf(" \t\t");
        end
    end
    fprintf("\n");
end

% Сведение задачи максимизации к задаче минимизации
function matr = convertToMinimizationProblem(matr)
    maxElem = max(max(matr));
    matr = matr * (-1) + maxElem;
end

% Нахождение наименьшего элемента в каждом столбце матрицы C
```

```
% и вычитание его из соответствующего столбца
function matr = updateColumns(matr)
    minElemArr = min(matr);
    for i = 1 : length(minElemArr)
        matr(:, i) = matr(:, i) - minElemArr(i);
    end
end

% Нахождение наименьшего элемента в каждой строке матрицы C
% и вычитание его из соответствующей строки
function matr = updateRows(matr)
    minElemArr = min(matr, [], 2);
    for i = 1 : length(minElemArr)
        matr(i, :) = matr(i, :) - minElemArr(i);
    end
end

% Построение начальной СНН
function matrSIZ = makeSIZ(matr)
    [numRows, numCols] = size(matr);
    matrSIZ = zeros(numRows, numCols);

    for i = 1 : numRows
        for j = 1 : numCols
            if matr(i, j) == 0
                count = 0;
                for k = 1 : numCols
                   count = count + matrSIZ(i, k);
                end
                for k = 1 : numRows
                   count = count + matrSIZ(k, j);
                end
                if count == 0
                    matrSIZ(i, j) = 1;
                end
            end
        end
    end
end
```

```
% Выделение столбцов, в которых стоит 0*
function [selection] = makeSelection(numRows, numCols,
   selectedColumns)
    selection = zeros(numRows, numCols);
    for j = 1 : numCols
        if selectedColumns(j) == 1
            selection(:, j) = selection(:, j) + 1;
        end
    end
end

% Изменить матрицу в случае, если среди невыделенных элементов нет
   нуля
function [matr] = updateMatrNoZero(matr, numRows, numCols,
   selection, selectedRows, selectedColumns)
    h = -1;
    for i = 1 : numCols
        for j = 1 : numRows
            if selection(j, i) == 0 && (matr(j, i) < h || h == -1)
                h = matr(j, i);
            end
        end
    end
    fprintf("h = %d\n", h);

    for i = 1 : numCols
        if selectedColumns(i) == 0
            matr(:, i) = matr(:, i) - h;
        end
    end
    for i = 1 : numRows
        if selectedRows(i) == 1
            matr(i, :) = matr(i, :) + h;
        end
    end
end

% Найти 0* в той же строке, что и 0'
```

```
function [zeroStarInRow] = getZeroStarInRow(streakPoint, numCols,
   matrSIZ)
    j = streakPoint(1);
    zeroStarInRow = [-1 -1];
    for i = 1 : numCols
       if matrSIZ(j, i) == 1
           zeroStarInRow(1) = j;
           zeroStarInRow(2) = i;
           break
       end
    end
end

% Построить L-цепочку
function [matrStreak, matrSIZ] = makeLChain(numRows, numCols,
   streakPoint, matrStreak, matrSIZ, debugMode)
    if debugMode == true
        fprintf("Построенная L-цепочка:");
    end

    i = streakPoint(1);
    j = streakPoint(2);
    while i > 0 && j > 0 && i <= numRows && j <= numCols
        % Снять '
        matrStreak(i, j) = 0;
        % Поставить *
        matrSIZ(i, j) = 1;

        if debugMode == true
            fprintf("[%d, %d] ", i, j);
        end

        % Дойти до 0* по столбцу от 0'
        row = 1;
        while row <= numRows && (matrSIZ(row, j) ~= 1 || row == i)
            row = row + 1;
        end

        if row <= numRows
```

```
            % Дойти до 0' по строке от 0*
            col = 1;
            while col <= numCols && (matrStreak(row, col) ~= 1 ||
               col == j)
                col = col + 1;
            end

            if col <= numCols
                matrSIZ(row,j) = 0;

                if debugMode == true
                    fprintf("-> [%d, %d] -> ", row, j);
                end
            end
            j = col;
        end
        i = row;
    end

    if debugMode == true
        fprintf("\n");
    end
end

function [fOpt] = getFOpt(matr, matrSIZ)
    fOpt = 0;
    [numRows, numCols] = size(matr);

    for i = 1 : numRows
        for j = 1 : numCols
            if matrSIZ(i, j) == 1
                fOpt = fOpt + matr(i, j);
            end
        end
    end
end
```

2.2 Результаты расчётов для задач из варианта №9

Листинг 2.2 – Решение задачи минимизации

```
Матрица стоимостей (9 вариант):
     4      7      1      5      5
     6      8      3      7      6
     6      4      5      7      7
     4      2      3      4      9
     8      1      8      3      8

Результат вычитания наименьшего элемента по столбцам:
     0      6      0      2      0
     2      7      2      4      1
     2      3      4      4      2
     0      1      2      1      4
     4      0      7      0      3

Результат вычитания наименьшего элемента по строкам:
     0      6      0      2      0
     1      6      1      3      0
     0      1      2      2      0
     0      1      2      1      4
     4      0      7      0      3

Начальная СНН:
    0*   6    0    2    0
    1    6    1    3    0*
    0    1    2    2    0
    0    1    2    1    4
    4    0*   7    0    3

Число нулей в построенной СНН: k = 3

k < n = 5 ==> СНН нужно улучшать
----------------------- Итерация 1 -----------------------

Результат выделения столбцов, в которых стоит 0*:
    0*       6        0        2        0
```

```
    1       6       1       3       0*
    0       1       2       2       0
    0       1       2       1       4
    4       0*      7       0       3
    +       +                       +

Среди невыделенных элементов есть 0, отметим 0':
    0*      6       0'      2       0
    1       6       1       3       0*
    0       1       2       2       0
    0       1       2       1       4
    4       0*      7       0       3
    +       +                       +

В одной строке с 0' есть 0*, перебросим выделение со столбца на
   строку:
+   0*      6       0'      2       0
    1       6       1       3       0*
    0       1       2       2       0
    0       1       2       1       4
    4       0*      7       0       3
            +                       +

Среди невыделенных элементов есть 0, отметим 0':
+   0*      6       0'      2       0
    1       6       1       3       0*
    0'      1       2       2       0
    0       1       2       1       4
    4       0*      7       0       3
            +                       +
Построенная L-цепочка:[3, 1] -> [1, 1] -> [1, 3]

В пределах L-цепочки 0* заменем на 0, а 0' на 0*:
    0   6   0*  2   0
    1   6   1   3   0*
    0*  1   2   2   0
    0   1   2   1   4
    4   0*  7   0   3
```

```
Число нулей в построенной СНН: k = 4

k < n = 5 ==> СНН нужно улучшать
----------------------- Итерация 2 -----------------------

Результат выделения столбцов, в которых стоит 0*:
    0       6       0*      2       0
    1       6       1       3       0*
    0*      1       2       2       0
    0       1       2       1       4
    4       0*      7       0       3
    +       +       +               +

Среди невыделенных элементов есть 0, отметим 0':
    0       6       0*      2       0
    1       6       1       3       0*
    0*      1       2       2       0
    0       1       2       1       4
    4       0*      7       0'      3
    +       +       +               +

В одной строке с 0' есть 0*, перебросим выделение со столбца на
   строку:
    0       6       0*      2       0
    1       6       1       3       0*
    0*      1       2       2       0
    0       1       2       1       4
+   4       0*      7       0'      3
    +               +               +

Среди невыделенных элементов нет 0, преобразуем матрицу:
h = 1
    0       5       0*      1       0
    1       5       1       2       0*
    0*      0       2       1       0
    0       0       2       0       4
+   5       0*      8       0'      4
    +               +               +
```

```
Среди невыделенных элементов есть 0, отметим 0':
    0       5       0*      1       0
    1       5       1       2       0*
    0*      0'      2       1       0
    0       0       2       0       4
+   5       0*      8       0'      4
    +               +               +

В одной строке с 0' есть 0*, перебросим выделение со столбца на
   строку:
    0       5       0*      1       0
    1       5       1       2       0*
+   0*      0'      2       1       0
    0       0       2       0       4
+   5       0*      8       0'      4
                    +               +

Среди невыделенных элементов есть 0, отметим 0':
    0'      5       0*      1       0
    1       5       1       2       0*
+   0*      0'      2       1       0
    0       0       2       0       4
+   5       0*      8       0'      4
                    +               +

В одной строке с 0' есть 0*, перебросим выделение со столбца на
   строку:
+   0'      5       0*      1       0
    1       5       1       2       0*
+   0*      0'      2       1       0
    0       0       2       0       4
+   5       0*      8       0'      4
                                    +

Среди невыделенных элементов есть 0, отметим 0':
+   0'      5       0*      1       0
    1       5       1       2       0*
+   0*      0'      2       1       0
    0'      0       2       0       4
```

```
+   5       0*      8       0'      4
                                    +
Построенная L-цепочка:[4, 1] -> [3, 1] -> [3, 2] -> [5, 2] -> [5,
   4]

В пределах L-цепочки 0* заменем на 0, а 0' на 0*:
    0   5   0*  1   0
    1   5   1   2   0*
    0   0*  2   1   0
    0*  0   2   0   4
    5   0   8   0*  4

Число нулей в построенной СНН: k = 5

Конечная СНН:
    0   5   0*  1   0
    1   5   1   2   0*
    0   0*  2   1   0
    0*  0   2   0   4
    5   0   8   0*  4

X:
     0      0      1      0      0
     0      0      0      0      1
     0      1      0      0      0
     1      0      0      0      0
     0      0      0      1      0

f_opt = 18
```

Листинг 2.3 – Решение задачи максимизации

```
Матрица стоимостей (9 вариант):
     4      7      1      5      5
     6      8      3      7      6
     6      4      5      7      7
     4      2      3      4      9
     8      1      8      3      8

Матрица стоимостей после сведения к задаче минимизации:
```

```
     5      2      8      4      4
     3      1      6      2      3
     3      5      4      2      2
     5      7      6      5      0
     1      8      1      6      1

Результат вычитания наименьшего элемента по столбцам:
     4      1      7      2      4
     2      0      5      0      3
     2      4      3      0      2
     4      6      5      3      0
     0      7      0      4      1

Результат вычитания наименьшего элемента по строкам:
     3      0      6      1      3
     2      0      5      0      3
     2      4      3      0      2
     4      6      5      3      0
     0      7      0      4      1

Начальная СНН:
    3    0*   6    1    3
    2    0    5    0*   3
    2    4    3    0    2
    4    6    5    3    0*
    0*   7    0    4    1

Число нулей в построенной СНН: k = 4

k < n = 5 ==> СНН нужно улучшать
----------------------- Итерация 1 -----------------------

Результат выделения столбцов, в которых стоит 0*:
    3         0*        6         1         3
    2         0         5         0*        3
    2         4         3         0         2
    4         6         5         3         0*
    0*        7         0         4         1
    +         +                   +         +
```

```
Среди невыделенных элементов есть 0, отметим 0':
    3       0*      6       1       3
    2       0       5       0*      3
    2       4       3       0       2
    4       6       5       3       0*
    0*      7       0'      4       1
    +       +               +       +

В одной строке с 0' есть 0*, перебросим выделение со столбца на
   строку:
    3       0*      6       1       3
    2       0       5       0*      3
    2       4       3       0       2
    4       6       5       3       0*
+   0*      7       0'      4       1
            +               +       +

Среди невыделенных элементов нет 0, преобразуем матрицу:
h = 2
    1       0*      4       1       3
    0       0       3       0*      3
    0       4       1       0       2
    2       6       3       3       0*
+   0*      9       0'      6       3
            +               +       +

Среди невыделенных элементов есть 0, отметим 0':
    1       0*      4       1       3
    0'      0       3       0*      3
    0       4       1       0       2
    2       6       3       3       0*
+   0*      9       0'      6       3
            +               +       +

В одной строке с 0' есть 0*, перебросим выделение со столбца на
   строку:
    1       0*      4       1       3
+   0'      0       3       0*      3
```

```
    0       4       1       0       2
    2       6       3       3       0*
+   0*      9       0'      6       3
            +                       +

Среди невыделенных элементов есть 0, отметим 0':
    1       0*      4       1       3
+   0'      0       3       0*      3
    0'      4       1       0       2
    2       6       3       3       0*
+   0*      9       0'      6       3
            +                       +
Построенная L-цепочка:[3, 1] -> [5, 1] -> [5, 3]

В пределах L-цепочки 0* заменем на 0, а 0' на 0*:
    1   0*  4   1   3
    0   0   3   0*  3
    0*  4   1   0   2
    2   6   3   3   0*
    0   9   0*  6   3

Число нулей в построенной СНН: k = 5

Конечная СНН:
    1   0*  4   1   3
    0   0   3   0*  3
    0*  4   1   0   2
    2   6   3   3   0*
    0   9   0*  6   3

X:
     0     1     0     0     0
     0     0     0     1     0
     1     0     0     0     0
     0     0     0     0     1
     0     0     1     0     0

f_opt = 37
```