**Отчёт по лабораторной работе №1**

**по дисциплине**

**Методы вычислений**

Выполнил: студент ИУ7-17

Пахомов А.А.

Вариант 1

МГТУ, 2015 г.

**Теоретическая часть**

**Содержательная постановка задачи о назначениях:**

В распоряжении работодателя имеется *n* работ и *n* исполнителей. Стоимость выполнения *i*-ой работы *j*-ым исполнителем составляет *сij > 0* единиц. Необходимо распределить все работы между исполнителями таким образом, чтобы:

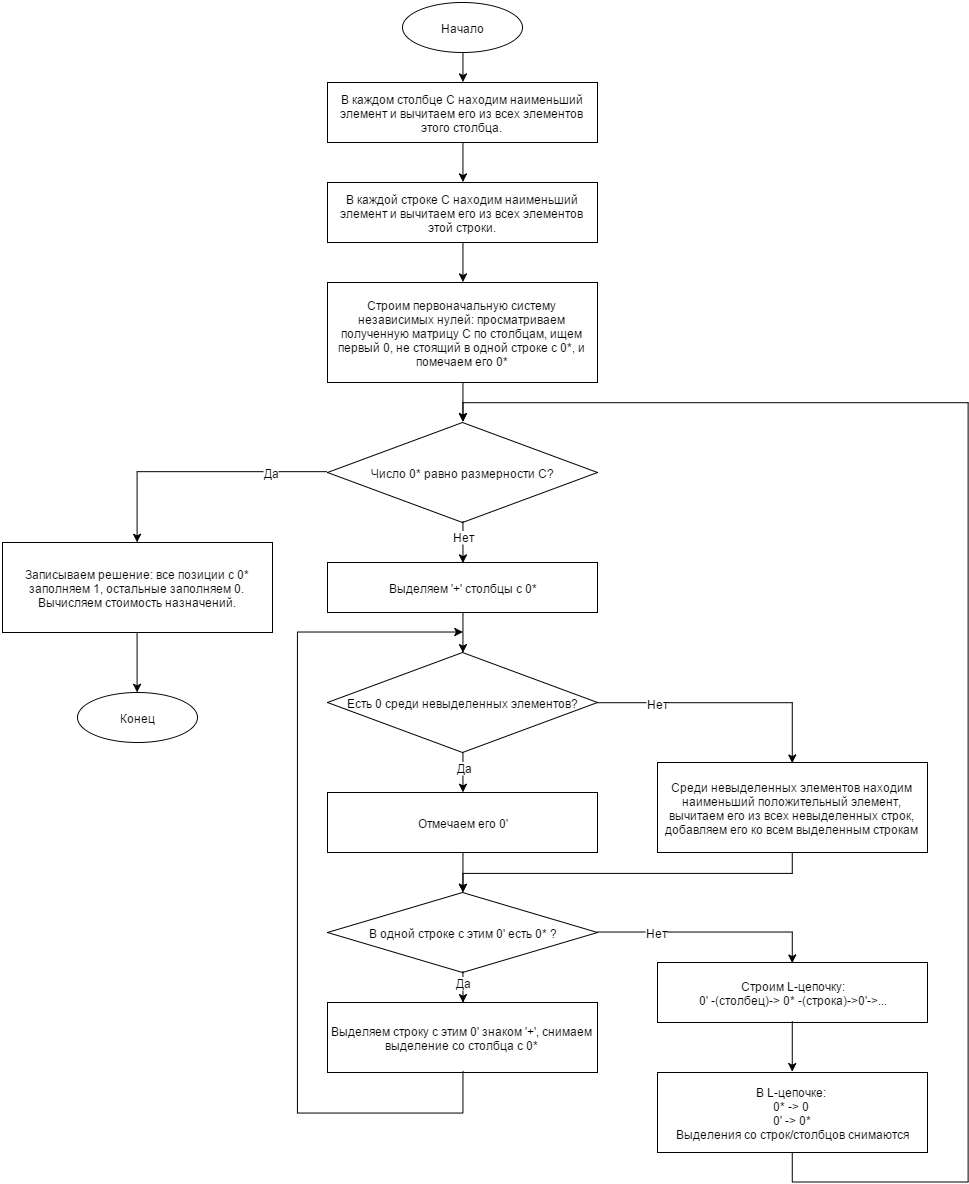
* Каждый исполнитель исполнял ровно одну работу
* Суммарная стоимость выполнения всех работ была бы минимальной

**Математическая постановка задачи о назначениях:**

**Входные данные:**

Входные данные задаются в виде матрицы стоимостей C, для которой требуется решить задачу о назначениях в форме задач минимизации и максимизации.

**Венгерский метод решения задачи о назначениях:**



**Текст программы**

*lab1.m*

m=dlmread('input.txt');

fprintf('Матрица стоимостей = \n');

disp(m);

% минимизация с отладочной печатью

[assignment,cost] = hungarian\_method(m, false, true);

fprintf('\nx(opt) = \n');

disp(assignment);

fprintf('\nf(opt) = %d\n\n', cost);

*hungarian\_metod.m*

function [assignment,cost] = hungarian\_method(matr, maximization, debugging)

costMat = matr;

if(maximization)

costMat = -matr + max(max(matr));

end;

assignment = false(size(costMat));

cost = 0;

costMat(costMat~=costMat)=Inf;

validMat = costMat<Inf;

validCol = any(validMat);

validRow = any(validMat,2);

nRows = sum(validRow);

nCols = sum(validCol);

n = max(nRows,nCols);

if ~n

return

end

dMat = zeros(n);

dMat(1:nRows,1:nCols) = costMat(validRow,validCol);

% Шаг 1. Вычтем минимальный элемент из каждой строки

dMat = bsxfun(@minus, dMat, min(dMat,[],2));

% Шаг 2. Для каждого нуля: если нет больше нулей в его строке или столбце, то

% отмечаем \*

zP = ~dMat;

starZ = false(n);

while any(zP(:))

[r,c]=find(zP,1);

starZ(r,c)=true;

zP(r,:)=false;

zP(:,c)=false;

end

k=1;

while 1

if(debugging)

% отладочная печать

fprintf('Итерация %d:\n', k);

k = k+1;

for i=1:n

for j=1:n

if(starZ(i,j) == true)

fprintf('\t0\*');

else

fprintf(' \t%d', dMat(i,j));

end;

end;

fprintf('\n');

end;

fprintf('\n');

end;

% Шаг 3. Отмечаем столбцы, в которых есть 0\*. Если все столбы отмечены, то решение

% найдено

primeZ = false(n);

coverColumn = any(starZ);

if ~any(~coverColumn)

break

end

coverRow = false(n,1);

while 1

% Шаг 4. Находим 0 в неотмеченных столбцах (строках) и отмечаем его

% Если в строке или столбце с ним нет 0\*, то переходим к шагу 5.

% Иначе снимаем выделение со столбца с 0\* и отмечаем строку с теку-

% щим 0'. Повторяем до тех пор, пока не останется неотмеченных

% нулей. Сохраняем наименьшее значение из неотмеченных

% столцов(строк) и переходим к шагу 6.

zP(:) = false;

zP(~coverRow,~coverColumn) = ~dMat(~coverRow,~coverColumn);

Step = 6;

while any(any(zP(~coverRow,~coverColumn)))

[uZr,uZc] = find(zP,1);

primeZ(uZr,uZc) = true;

stz = starZ(uZr,:);

if ~any(stz)

Step = 5;

break;

end

coverRow(uZr) = true;

coverColumn(stz) = false;

zP(uZr,:) = false;

zP(~coverRow,stz) = ~dMat(~coverRow,stz);

end

if Step == 6

% Шаг 6. Вычитаем сохранённое значение из каждого непокрытого

% столбца и добавляем к каждой покрытой строке. Возвращаемся к

% шагу 4.

M=dMat(~coverRow,~coverColumn);

minval=min(min(M));

if minval==inf

return

end

dMat(coverRow,coverColumn)=dMat(coverRow,coverColumn)+minval;

dMat(~coverRow,~coverColumn)=M-minval;

else

break

end

end

% Шаг 5. Строим L-цепочку.

rowZ1 = starZ(:,uZc);

starZ(uZr,uZc)=true;

while any(rowZ1)

starZ(rowZ1,uZc)=false;

uZc = primeZ(rowZ1,:);

uZr = rowZ1;

rowZ1 = starZ(:,uZc);

starZ(uZr,uZc)=true;

end

end

% Результаты

assignment(validRow,validCol) = starZ(1:nRows,1:nCols);

cost = sum(matr(assignment));

**Результаты вычислений**

**Задача минимизации:**

Матрица стоимостей =

4 2 1 3 7

1 5 4 6 3

5 4 8 7 2

9 9 3 2 5

3 4 7 8 2

x(opt) =

0 0 1 0 0

1 0 0 0 0

0 0 0 0 1

0 0 0 1 0

0 1 0 0 0

f(opt) = 10

**Задача максимизации:**

Матрица стоимостей =

4 2 1 3 7

1 5 4 6 3

5 4 8 7 2

9 9 3 2 5

3 4 7 8 2

x(opt) =

0 0 0 0 1

0 1 0 0 0

0 0 1 0 0

1 0 0 0 0

0 0 0 1 0

f(opt) = 37