ПРИМЕНЕНИЕ MS EXCEL ПРИ НАХОЖДЕНИИ ОПТИМАЛЬНОГО МЕСТА РАСПОЛОЖЕНИЯ СКЛАДА

Н. Н. Рачковский
Государственный институт управления и социальных технологий БГУ,
Минск, Беларусь

Предположим, что имеется n населенных пунктов $C_1, C_2, \dots, C_n$, причем некоторые из них связаны между собой дорогами определенной протяженности. В каждом из этих населенных пунктов находится торговая точка. Требуется среди этих пунктов выбрать наиболее подходящий для расположения в нем склада товара, предназначенного для продажи в указанных точках. Другими словами, нужно среди населенных пунктов $C_1, C_2, \dots, C_n$ найти такой C_{optim}, чтобы суммарная протяженность всех дорог, соединяющих этот C_{optim} с остальными пунктами, была минимальной (предполагается, что поставка товара со склада в каждую торговую точку осуществляется отдельным рейсом).

Для решения этой задачи предлагается использовать матрицу кратчайших расстояний между населенными пунктами, элементами являются протяженности кратчайших путей, соединяющих соответствующие населенные пункты. Для построения этой матрицы предлагается использовать конечный итерационный процесс, суть которого в следующем. На первом этапе процесса рассматривается матрица

$$D = \begin{pmatrix} d_{11} & d_{12} & \dots & d_{1n} \\ d_{21} & d_{22} & \dots & d_{2n} \\ \dots & \dots & \dots & \dots \\ d_{n1} & d_{n2} & \dots & d_{nn} \end{pmatrix}$$

протяженностей имеющихся дорог: значение d_{ij} равно длине дороги, непосредственно соединяющей населенные пункты C_i и C_j, если такая дорога есть, и нулю в противном случае; кроме того, все элементы d_{ii}, стоящие на главной диагонали этой матрицы, равны нулю (предполагается, что каждый населенный пункт C_i соединен с самим собой дорогой нулевой протяженности). Отметим следующие очевидные свойства матрицы D_0. Во-первых, эта матрица квадратна; во-вторых она является симметрической, т. е. для всех значений i и j справедливо равенство $d_{ij} = d_{ji}$. Действительно, наличие дороги между пунктами C_i и C_j означает наличие дороги между пунктами C_j и C_i той же протяженности. В-третьих, каждая i-я строка матрицы D_0 состоит из протяженностей дорог, выходящих из соответствующего населенного пункта C_i; то же самое можно сказать и о каждом столбце.

На второй итерации для каждой пары пунктов (C_i, C_j) находятся все пункты C_k, непосредственно соединенные дорогами с обоими пунктами C_i и C_j, и для каждого такого «транзитного» пункта C_k вычисляется протяженность пути из пункта C_i в пункт C_j через пункт C_k; затем среди всех таких путей выбирается путь минимальной протяженности d_{ij}^1. В результате получаем уточненную матрицу

$$D_1 = \begin{pmatrix} d_{11}^1 & d_{12}^1 & \dots & d_{1n}^1 \\ d_{21}^1 & d_{22}^1 & \dots & d_{2n}^1 \\ \dots & \dots & \dots & \dots \\ d_{n1}^1 & d_{n2}^1 & \dots & d_{nn}^1 \end{pmatrix}$$

расстояний между населенными пунктами. Для построения этой матрицы рассмотрим матрицу

$$R_0 = \begin{pmatrix} r_{11} & r_{12} & \dots & r_{1n} \\ r_{21} & r_{22} & \dots & r_{2n} \\ \dots & \dots & \dots & \dots \\ r_{n1} & r_{n2} & \dots & r_{nn} \end{pmatrix}$$

имеющихся дорог. Элементы матрицы R_0 вычисляются по матрице D_0: $r_{ij} = \begin{cases} 1, \text{если } d_{ij} \neq 0, \\ 0, \text{если } d_{ij} = 0 \end{cases}$ при $i \neq j$;

все элементы r_{ii}, стоящие на главной диагонали матрицы R_0, равны 1. Другими словами, элементы матрицы R_0 указывают на наличие или отсутствие дороги, непосредственно соединяющей каждую пару пунктов (C_i, C_j): в случае наличия такой дороги соответствующий элемент r_{ij} равен 1, а в случае отсутствия – нулю. Заметим, что, как и матрица D_0, матрица R_0 является квадратной (тех же размеров) и симметрической. Для каждой пары населенных пунктов (C_i, C_j) (для определенности будем считать, что $i < j$) вычислим величины $p_1 = r_{j1} \cdot d_{i1} + r_{i1} \cdot d_{j1}$, $p_2 = r_{j2} \cdot d_{i2} + r_{i2} \cdot d_{j2}, \dots, p_n = r_{jn} \cdot d_{in} + r_{in} \cdot d_{jn}$.

Заметим, что $p_i = r_{ji} \cdot d_{ii} + r_{ii} \cdot d_{ji} = r_{ji} \cdot 0 + 1 \cdot d_{ji} = d_{ji} = d_{ij}$; $p_j = r_{jj} \cdot d_{ij} + r_{ij} \cdot d_{jj} = 1 \cdot d_{ij} + r_{ij} \cdot 0 = d_{ij}$.

Другими словами, если между рассматриваемыми пунктами существует дорога, то среди соответствующих величин $p_1, p_2, \dots, p_n$ присутствует (причем дважды) протяженность d_{ij} этой дороги. Предположим теперь, что некоторый населенный пункт C_k непосредственно связан дорогами с обоими пунктами C_i и C_j; тогда соответствующая величина p_k будет равна:

$$p_k = r_{jk} \cdot d_{ik} + r_{ik} \cdot d_{jk} = 1 \cdot d_{ik} + 1 \cdot d_{jk} = d_{ik} + d_{jk},$$

т. е. среди величин $p_1, p_2, \dots, p_n$ присутствуют длины всех путей, ведущих из пункта C_i в пункт C_j через третьи пункты (если такие пути есть). Если же на-

селенный пункт C_k непосредственно не связан дорогой с пунктом C_i, то соответствующая величина p_k будет равна:

$$p_k = r_{jk} \cdot d_{ik} + r_{ik} \cdot d_{jk} = r_{jk} \cdot 0 + 0 \cdot d_{jk} = 0.$$

Аналогично показывается, что и в случае, когда населенный пункт C_k непосредственно не связан дорогой с пунктом C_j, соответствующая величина p_k будет равна нулю. Резюмируя вышесказанное, получаем, что если существует путь, ведущий из пункта C_i в пункт C_j либо непосредственно, либо через какой-нибудь третий пункт, то среди величин p_1, p_2,..., p_n обязательно присутствует протяженность этого пути; в противном же случае все величины p_1, p_2, ... , p_n равны нулю. Следовательно, обозначив

$$d_{ij}^1 = \begin{cases} \min\{p_k | k = 1; 2; \ldots; n, p_k \neq 0\}, \text{если существует } p_k \neq 0, \\ 0, \text{если все } p_k = 0, \end{cases}$$

получим, что величина d_{ij}^1 равна протяженности кратчайшего пути, ведущего из пункта C_i в пункт C_j либо непосредственно, либо через какой-нибудь третий пункт, если такие пути есть, и нулю в противном случае. Напомним, что величины d_{ij}^1 нами определены в случае, когда $i < j$. Для всех $i > j$ определим $d_{ij}^1 = d_{ji}^1$; кроме того, для $i = j$ определим $d_{ii}^1 = 0$.

Таким образом, матрица D_1, составленная из определенных выше элементов d_{ij}^1, действительно является уточненной (по сравнению с D_0) матрицей расстояний между населенными пунктами.

Для матрицы D_1 аналогичным образом можно построить уточняющую ее матрицу D_2, для которой, в свою очередь, можно построить уточняющую ее матрицу D_3, и т. д. Вполне очевидно, что этот процесс построения последовательности уточняющих друг друга матриц расстояний между населенными пунктами нужно остановить тогда, когда очередная матрица D_l полностью совпадет с предыдущей матрицей D_{l-1}. В этом случае последняя в построенной последовательности матрица D_l будет окончательной матрицей кратчайших расстояний (по имеющимся дорогам) между рассматриваемыми населенными пунктами C_1, C_2, ... , C_n.

Предположим, что мы уже вычислили матрицу кратчайших расстояний D_l. Каждая i-я строка этой матрицы состоит из протяженностей d_{ij}^l кратчайших путей из соответствующего населенного пункта C_i во все остальные пункты. Следовательно, для определения оптимального места расположения склада достаточно найти строку с минимальной суммой стоящих в ней элементов.

Таков в общих чертах предлагаемый в данной работе способ решения рассматриваемой логистической задачи. Сделаем несколько важных замечаний. Во-первых, отметим, что матрица кратчайших расстояний D_l будет получена за *конечное* число итераций. Действительно, процесс вычисления матрицы D_l можно рассматривать как процесс нахождения кратчайшего пути между двумя самыми отдаленными (по количеству расположенных между ними «транзитных» пунктов) населенными пунктами, причем этот путь «прокладывается» одновременно из всех «транзитных» пунктов во встречных направлениях. В наихудшем случае этот путь будет проходить через все населенные пункты C_1, C_2, ... , C_n, и, значит, он будет состоять из $n - 1$ дорог. В результате первой итерации будут найдены фрагменты искомого кратчайшего пути, состоящие из двух дорог; в результате второй итерации – фрагменты, состоящие уже из четырех дорог; в результате третьей итерации – из восьми и т. д. Следовательно, последняя матрица D_{l-1}, отличающаяся от предыдущей матрицы, будет получена на итерации с номером $l - 1$, для которого справедливо неравенство $2^{l-1} \geq n - 1$, причем значение l должно быть *минимальным* натуральным числом, удовлетворяющим этому неравенству. Отсюда получаем, что количество итераций l определяется равенством $l = \lceil \log_2(n - 1) \rceil + 1$ и потому в силу конечности числа n само является конечным числом. Здесь запись $\lceil \log_2(n - 1) \rceil$ обозначает округление до ближайшего большего целого $\log_2(n - 1)$. Так, при $n = 9$ имеем $\lceil \log_2(n - 1) \rceil = \lceil \log_2 8 \rceil = \lceil 3 \rceil = 3$, а при $n = 10$ имеем $\lceil \log_2(n - 1) \rceil = \lceil \log_2 9 \rceil = \lceil 3{,}1699 \rceil = 4$.

Во-вторых, для построения на очередной m-й итерации уточненной матрицы расстояний D_{m+1} нужно сначала построить разведанных путей R_m, элементы которой равны единице, если уже найден путь, соединяющий соответствующие населенные пункты, и нулю в противном случае. Следовательно, наличие в последней матрице разведанных путей R_l нулевых элементов означает, что некоторые населенные пункты являются недостижимыми друг для друга по имеющимся дорогам. В этом случае возникает отдельная задача нахождения оптимального варианта постройки недостающей дороги (или дорог).

В-третьих, заметим, что на каждой m-й итерации вычисление величин $p_k = r_{jk}^m \cdot d_{ik}^m + r_{ik}^m \cdot d_{jk}^m$ является процедурой несложной, но трудоемкой из-за большого их количества. Действительно, для каждой пары индексов (i, j), $i < j$ таких величин будет n; указанных же пар индексов $^1/_2 n \cdot (n - 1)$; следовательно, на каждой итерации придется вычислить $^1/_2 n^2 \cdot (n - 1)$ величин p_k. Конечно, для облегчения вычислительной работы можно использовать MS Excel, но и в этом случае непосредственный набор указанных формул может занять достаточно много времени. В связи с этим отметим, что применение функции умножения матриц МУМНОЖ в значительной степени упрощает процедуру вычисления величин p_k. Выясним, перемножением каких матриц можно получить величины p_k. Нетрудно видеть, что при вычис-

лении каждого значения p_k используются элементы матриц D_m и R_m, однако непосредственная проверка показывает, что ни в каком из произведений этих матриц эти значения не содержатся. Ситуация изменится, если мы рассмотрим матрицу

$$R_{ij}^m = \begin{pmatrix} 0 & \dots & 0 & r_{1j}^m & 0 & \dots & 0 & r_{1i}^m & 0 & \dots & 0 \\ 0 & \dots & 0 & r_{2j}^m & 0 & \dots & 0 & r_{2i}^m & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & r_{nj}^m & 0 & \dots & 0 & r_{ni}^m & 0 & \dots & 0 \end{pmatrix},$$

в которой i-й столбец совпадает с j-м столбцом матрицы R_m, j-й столбец совпадает с i-м столбцом матрицы R_m, а все остальные элементы равны нулю. Нетрудно убедиться, что на главной диагонали произведения $R_{ij}^m \cdot D_m$ как раз и будут находиться нужные нам величины $p_1, p_2, \dots, p_n$.

Из всего вышесказанного получаем следующий алгоритм решения рассматриваемой задачи.

1. На основе атласа дорог строим матрицу D_0 протяженностей имеющихся дорог между рассматриваемыми населенными пунктами.

2. Дальнейшие действия являются общими для всех итераций. Будем предполагать, что на m-й итерации уже построена матрица D_m. По матрице D_m строим соответствующую матрицу R_m.

3. По матрице R_m для каждой пары индексов (i, j), $i < j$ строим матрицу R_{ij}^m и вычисляем произведение матриц $R_{ij}^m \cdot D_m$.

4. Для каждого вычисленного произведения матриц $R_{ij}^m \cdot D_m$ определим величину d_{ij}^{m+1} как минимальный ненулевой элемент на главной диагонали матрицы $R_{ij}^m \cdot D_m$, если такие есть; если же вся главная диагональ матрицы $R_{ij}^m \cdot D_m$ состоит из нулей, то определим $d_{ij}^{m+1} = 0$.

5. Из найденных величин d_{ij}^{m+1} составим матрицу D_{m+1}: все элементы на главной диагонали матрицы D_{m+1} будем считать равными нулю; над главной диагональю расставим соответствующие величины d_{ij}^{m+1}; элементы под главной диагональю определим равенствами $d_{ij}^{m+1} = d_{ji}^{m+1}$.

6. Вычислим разность матриц $D_{m+1} - D_m$; если разность содержит ненулевые элементы, то переходим к пункту 2 данного алгоритма и проделаем все указанные действия уже для матрицы D_{m+1}; в противном случае перейдем к пункту 7.

7. Для последней найденной матрицы D_{m+1} вычислим суммы элементов в каждой строке; строка с минимальной такой суммой соответствует искомому населенному пункту C_{optim}, в котором расположение склада товара будет оптимальным.

Проиллюстрируем предложенный алгоритм примером и покажем, как при этом можно использовать MS Excel.

Для наглядности рассмотрим следующую простую схему дорог (рисунок).

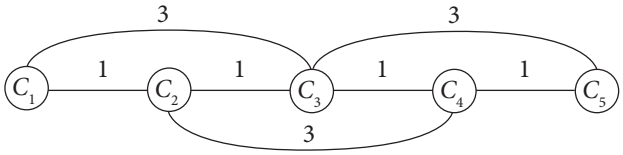


Рисунок – **Схема дорог к населенным пунктам**

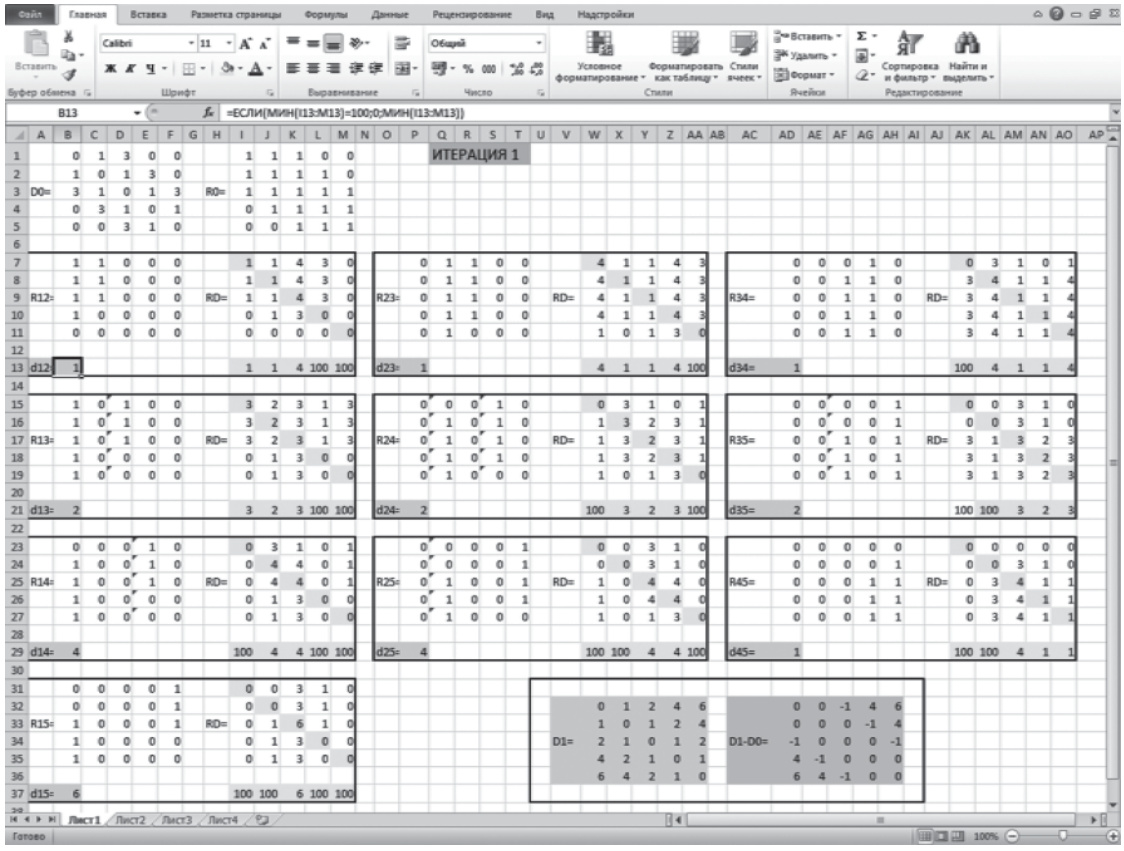
Ниже приведены расчеты в MS Excel согласно указанному выше алгоритму. Цветом помечены диагональные элементы матриц $R_{ij}^m \cdot D_m$, среди которых выбирается минимальное ненулевое значение. Для этого под каждой матрицей $R_{ij}^m \cdot D_m$ в желтой строке записаны все ненулевые элементы главной диагонали этой матрицы, а вместо нулевых элементов записано число 100 – число, большее суммарной протяженности всех имеющихся дорог. Оранжевым цветом помечена следующая матрица D_{m+1}, а зеленым – разность матриц $D_{m+1} - D_m$. Заметим, что положительные элементы этой разности указывают, что найден путь между соответствующими населенными пунктами, а отрицательные – что найден более короткий путь.

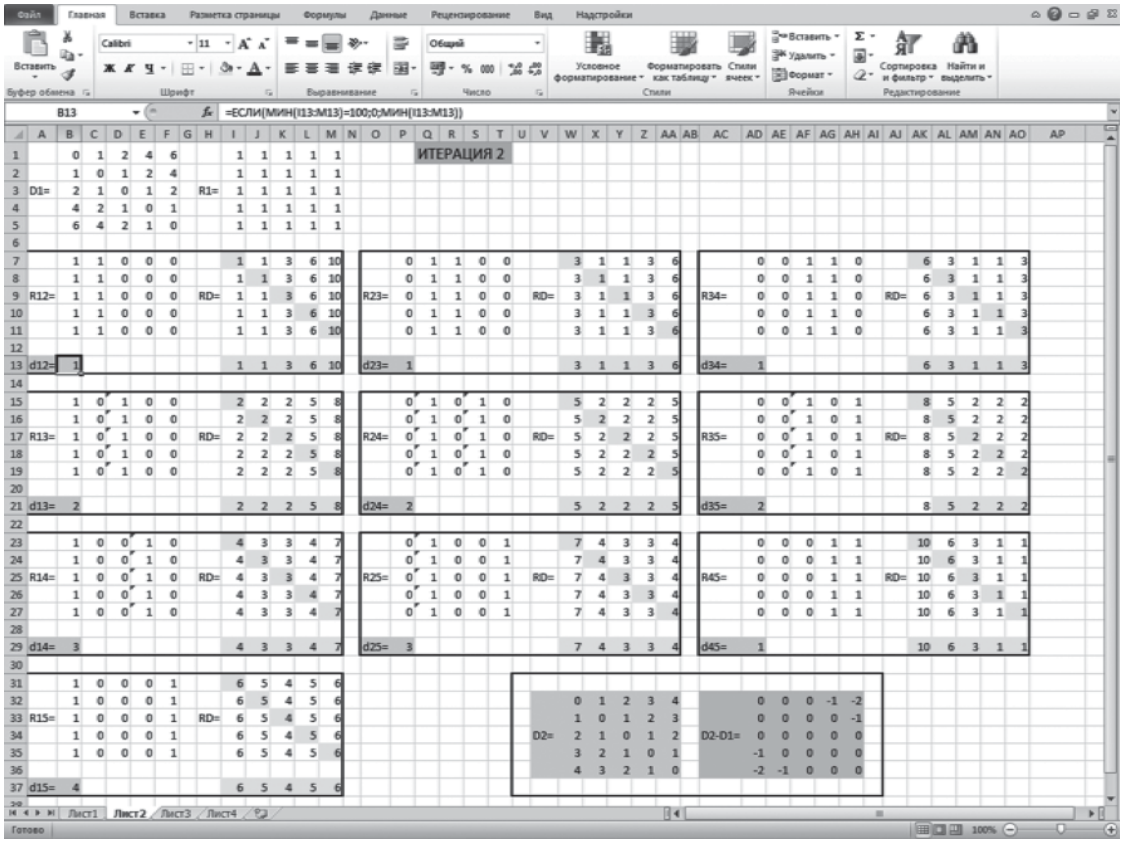
Таким образом, разность $D_3 - D_2$ является нулевой матрицей, следовательно, матрица D_3 (а значит, и матрица D_2) является матрицей кратчайших расстояний между населенными пунктами C_1, C_2, C_3, C_4, C_5. Сумма элементов первой строки матрицы D_3 равна $0+1+2+3+4=10$, для второй строки эта сумма равна $1+0+1+2+3=7$, для третьей, четвертой и пятой соответственно $2+1+0+1+2=6$, $3+2+1+0+1=7$ и $4+3+2+1+0=10$. Следовательно, оптимальным местом расположения склада товара будет населенный пункт C_3.

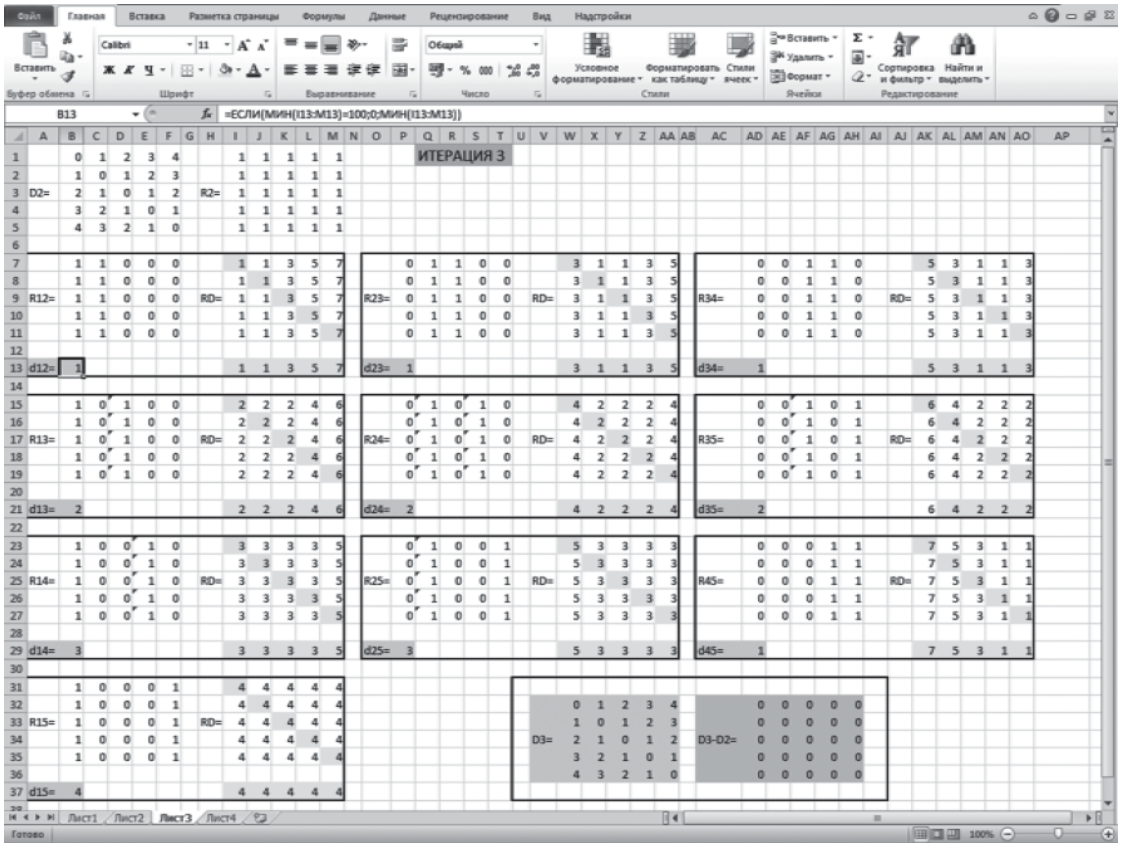
Заметим, что все используемые формулы достаточно ввести только один раз – на первой итерации, затем эти формулы можно перенести на остальные итерации. Предложенный выше алгоритм решения рассматриваемой задачи представляется более простым, чем алгоритмы, основанные на методах динамического программирования (см., например, [1]).

Литература

1. **Кузнецов, А. В.** Высшая математика: Математическое программирование / А. В. Кузнецов, В. А. Сакович, Н. И. Холод. – Минск : Вышэйшая школа, 2001. – 447 с.


ИТЕРАЦИЯ 1
=ЕСЛИ(МИН(I13:M13)=100;0;МИН(I13:M13))


ИТЕРАЦИЯ 2
=ЕСЛИ(МИН(I13:M13)=100;0;МИН(I13:M13))


Файл
Главная
Вставка
Разметка страницы
Формулы
Данные
Рецензирование
Вид
Надстройки
Вставить
Буфер обмена
Calibri
11
Шрифт
Выравнивание
Общий
Число
Условное форматирование
Форматировать как таблицу
Стили ячеек
Стили
Вставить
Удалить
Формат
Ячейки
Сортировка и фильтр
Найти и выделить
Редактирование
B13
=ЕСЛИ(МИН(I13:M13)=100;0;МИН(I13:M13))
ИТЕРАЦИЯ 3
R12=
R23=
R34=
R13=
R24=
R35=
R14=
R25=
R45=
R15=
D3=
D3-D2=
d12=
d23=
d34=
d13=
d24=
d35=
d14=
d25=
d45=
d15=
Лист1
Лист2
Лист3
Лист4
Готово
100%