УДК 519.85+519.1

И.В. ЛЫСЕНКО

Национальный аэрокосмический университет имени Н.Е. Жуковского «ХАИ», Харьков, Украина

О РЕШЕНИИ ЗАДАЧ ДИСКРЕТНОЙ ОПТИМИЗАЦИИ В СИСТЕМЕ КОМПЬЮТЕРНОЙ МАТЕМАТИКИ MATLAB

Рассматриваются возможности решения задач дискретной оптимизации в системе компьютерной математики Matlab. Описываются разработанные в среде Matlab встроенные функции для решения некоторых задач комбинаторной оптимизации (0-1-задачи о рюкзаке, задачи о покрытии множества, задачи коммивояжёра, задачи о назначении), а также приводятся примеры решения оптимизационных задач теории графов, допускающие интерпретацию в виде задач булевого линейного программирования, к числу которых относятся: задача о максимальном паросочетании, задача о минимальном вершинном покрытии, задача о минимальном рёберном покрытии, задача о максимальном независимом множестве вершин .

Ключевые слова: *дискретная оптимизация, задачи булевого линейного программирования.*

Введение

В практике принятия решений в самых разных областях человеческой деятельности приходится сталкиваться с задачами, относящимися к классу задач дискретной оптимизации. Существует множество методов, алгоритмов и программных средств решения этих задач. В этой связи справедливо ожидать возможности решения данных задач системами компьютерной математики (СКМ) – специализированными программными пакетами решения математических задач самого разного характера. К числу наиболее популярных СКМ относятся пакеты Mathematica, Maple, Mathcad, Matlab.

Так, система Matlab, имея мощный набор средств для решения разнообразных задач непрерывной оптимизации в виде пакетов Optimization Toolbox и Global Optimization Toolbox, не содержит встроенных функций для решения задач комбинаторной оптимизации [1, 2], а лишь – встроенную функцию *bintprog* для решения задачи булевого линейного программирования (БЛП) методом ветвей и границ. Некоторые исследователи стремятся восполнить этот пробел путём разработки соответствующих функций. Так, например, система Matlog [3], являющаяся расширением Matlab, содержит встроенные функции для решения некоторых задач дискретной оптимизации, интерпретируемых как задачи теории графов, а именно: задача коммивояжёра, задача нахождения кратчайшего пути графа, задача отыскания потока сети минимальной стоимости, задача нахождения минимального остовного дерева.

В [2] дано описание пакета Graph Theory Toolbox, разработанного харьковчанином профессором С.П. Иглиным. В данном пакете представлены функции для решения таких задач, как несимметричная задача коммивояжёра, задача нахождения максимального потока в сети, задача о максимальном паросочетании, задача отыскания минимального остовного дерева, задача нахождения минимального вершинного покрытия графа и др. Функции данного пакета используют в свою очередь функции расширения TOMLAB [4] для решения задач целочисленного и смешанного квадратичного программирования в виду того, что, как замечается в [1], «многие задачи на графах могут быть сформулированы в терминах целочисленного линейного программирования (ЦЛП)» и «фактически проблема сводится к тому, чтобы информацию о графе сформулировать как задачу ЦЛП, решить её и вернуть результаты в терминах теории графов».

Целью статьи является описание реализованных в виде m-файлов (файлов системы Matlab) функций для решения некоторых задач комбинаторной оптимизации, сводимых к задачам БЛП (в частности, задачи коммивояжёра, задачи о назначении, задачи о покрытии множества, 0,1-задачи о рюкзаке), а также иллюстрация использования встроенной функции *bintprog* для решения оптимизационных задач теории графов.

1. Решение экстремальных комбинаторных задач

Согласно [5], экстремальные комбинаторные задачи – это задачи, в которых элементами множества решений являются перестановки из n символов

(объектов). К числу таких задач относятся задача о назначении и задача коммивояжёра.

Что касается *задачи о назначении*, то, как известно, её суть состоит в отыскании наилучшего – в смысле минимальной стоимости – распределения n работников по n работам при известных затратах c_{ij}, связанных с назначением работника с номером i на работу с номером j (i,j = 1, … , n). Формальная постановка данной задачи имеет вид:

$$f = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \to \min, \qquad (1)$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j = \overline{1,n}; \quad \sum_{j=1}^{n} x_{ij} = 1 \ \forall i = \overline{1,n}. \qquad (2)$$

При этом $x_{ij} \in \{0,1\}$ и $x_{ij} = 1$, если i-й работник назначается на j-ю работу и $x_{ij} = 0$ в противном случае.

М-файл-функция с именем *assignprob* для решения задачи о назначении имеет вид:

```
function [] = assignprob(f)
     n = sqrt(length(f));
     A1 = genMatrixA1(n);
     A2 = repmat(eye(n),1,n);
     A = [A1;A2];
     b = ones(2*n,1);
    [x,fval] = bintprog(f, [], [], A, b);
x
disp('Значение целевой функции')
fval
end
  function A = genMatrixA1(n)
k = 1;
for i = 1:n
   for ii = 1:n * n
      if(k>(i-1)*n && k<((i-1)*n)+n+1)
         A(i,ii) = 1;
      else
         A(i,ii) = 0;
      end;
         k = k + 1;
      end;
        k = 1;
    end;
end.
```

Входным параметром f данной функции является вектор (столбец) значений c_{ij}. В теле данной функции используется функция *genMatrixA1* для формирования части системы ограничений (2).

Следует заметить, что задача о назначении, которая в терминах теории графов определяется как задача поиска наименьшего взвешенного паросочетания в двудольном полносвязном графе с равными долями, может быть решена путём несложной модификации Matlab-функции *grMaxMatch* пакета Graph Theory Toolbox [2], позволяющей решать задачу отыскания максимального взвешенного паросочетания.

Что касается *задачи коммивояжёра*, то, пользуясь терминологией теории графов, она заключается в отыскании минимального по длине гамильтонова цикла в графе.

Формальная постановка данной задачи имеет вид:

$$f = \sum_{i=1}^{n}\sum_{\substack{j=1 \\ i\neq j}}^{n} c_{ij}x_{ij} \to \min, \qquad (3)$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j = \overline{1,n}; \quad \sum_{j=1}^{n} x_{ij} = 1 \ \forall i = \overline{1,n}. \qquad (4)$$

Здесь c_{ij} – расстояние между i-й и j-й вершиной в графе; $x_{ij} \in \{0,1\}$ и $x_{ij} = 1$, если путь проходит из i-й вершины в j-ю и $x_{ij} = 0$ в противном случае.

М-файл-функция с именем tsp для решения несимметричной задачи коммивояжёра имеет вид:

```
function [] = tsp(f)
[s1,s2] = size(f);
i = 1; sq = 1;
while (sq < s1)
    sq = i * i;
    if (sq >= s1)
        sq = i; break;
    end;
    i = i + 1;
end;
n = sq; A = constraintsmatrixA(n);
b = ones(2*n,1);
[x,fval] = bintprog(f, [], [], A, b);
x
disp('Значение целевой функции')
fval
end
function [m] = constraintsmatrixA(n)
k = 1; z = 1;
for i = 1:n
   for ii = 1:n
       if(k == i)
       else
           tmp(i,ii) = z;
           z = z + 1;
       end;
       k = k + 1;
   end;
   k = 1;
end;
k = 1;
for i = 1:n
   for ii = 1:n
       if(k == i)
       else
           c(i,tmp(ii, i)) = 1;
       end;
       k = k + 1;
   end;
   k = 1;
end;
m = vertcat(matrixA2(n), c);
end
function A = matrixA2(n)
k = 1;
for i = 1:n
  for ii = 1:n *(n-1)
```

```
    if(k > (n-1)*(i-1) && k < (n-1)*(i-1)+n)
       A(i,ii) = 1;
         else
           A(i,ii) = 0;
         end;
           k = k + 1;
         end;
           k = 1;
     end;
end.
```

Входным параметром f данной функции является вектор (столбец) значений c_{ij}. В теле данной функции используются функции *matrixA2* и *constraintsmatrixA* для формирования ограничений (4), причём функция *matrixA2* является частью функции *constraintsmatrixA*.

В связи с разработанной функцией для решения задачи коммивояжёра следует отметить, что ограничениям (4) соответствует система из нескольких меньших циклов (подциклов), которые в сумме охватывают все вершины. Чтобы устранить подциклы и всегда получать в качестве решения гамильтонов цикл, необходимо ввести дополнительные ограничения, в которых присутствуют неограниченные действительные переменные, ассоциированные с вершинами. Однако реализовать эти ограничения, оставаясь в рамках функции *bintprog*, невозможно, поэтому при использовании функции *bintprog* решение задачи коммивояжёра может быть представлено набором подциклов.

В этой связи преимущество функции *grTravSale* пакета Graph Theory Toolbox [2] для решения несимметричной задачи коммивояжёра очевидно.

В то же время, следует заметить, что в тех случаях, когда обе функции (*tsp* и *grTravSale*) дают один и тот же корректный результат, его нахождение с помощью функции *tsp* осуществляется значительно быстрее, чем с помощью функции *grTravSale* для графа с числом вершин, большим 6. Так, например, для графа с числом вершин, равным 8 и матрицей расстояний, элементы которой есть числа в диапазоне от 6 до 11, время получения результата на основе функции *grTravSale* равно 15,05 сек., а на основе функции *tsp* – 0,046 сек.

2. Решение задач о рюкзаке и покрытии множества

Что касается *0,1-задачи о рюкзаке*, то её суть, как известно, состоит в отыскании такого набора предметов из заданного их множества с размерностью a_i и стоимостью c_i каждого, которые, будучи помещёнными в рюкзак размерности A, обеспечивали бы максимальную стоимость. Формальная постановка данной задачи имеет вид:

$$f = \sum_{i=1}^{n} c_i x_i \to \max, \tag{5}$$

$$\sum_{i=1}^{n} a_i x_i \le A. \tag{6}$$

При этом $x_i \in \{0,1\}$ и $x_i = 1$, если i-й предмет кладётся в рюкзак и $x_i = 0$ в противном случае.

М-файл-функция с именем *knapsack01* для решения 0,1-задачи о рюкзаке имеет вид:

```
function func = knapsack01(f, A, b)
  [x,fval]= bintprog(-f, A, b);
   x
   fval = -fval
 end.
```

Входными параметрами данной функции являются: f – вектор (столбец) значений c_i; A – вектор значений a_i; b – размерность рюкзака.

Сущность задачи о покрытии множества, как известно, заключается в следующем. Имеется некоторое множество $M = \{h_1, \dots, h_m\}$ объектов h_i ($i = 1, \dots, m$), а также $S = \{S_1, \dots, S_n\}$ – семейство подмножеств S_j ($j = 1, \dots, n$), содержащих элементы множества M, и каждому из этих подмножеств поставлено в соответствие некоторое число (вес) c_j. Требуется найти такой набор подмножеств $S^* \subset S$, при котором достигается покрытие множества M с минимальным суммарным весом.

Формально данная задача может быть представлена в виде:

$$f = \sum_{j=1}^{n} c_j x_j \to \min, \tag{7}$$

$$\sum_{j=1}^{n} a_{ij} x_j \ge 1, \; i = 1, \dots, m. \tag{8}$$

Здесь $x_j \in \{0,1\}$ и $x_j = 1$, если подмножество S_j входит в покрытие множества M, и $x_j = 0$ в противном случае; $a_{ij} \in \{0,1\}$ и $a_{ij} = 1$, если $h_i \in S_j$ и $a_{ij} = 0$ в противном случае.

М-файл-функция с именем *covering* для решения задачи о покрытии множества имеет вид:

```
function [x, fval] = covering(f,s)
n = size (f,1); z = 0;
for i=1:n
    b=max(s{i});
    z=max(z,b);
end
A = zeros(z, n);
for i = 1:n
    for j = 1:size(s{i},2)
        A(s{i}(j),i)=1;
    end
end
b = ones(z,1);
[x, fval] = bintprog(f, -A, -b).
```

Входными параметрами данной функции являются: f – вектор (столбец) значений c_i; s – набор векторов s{j} (j = 1,…,n), содержащих элементы соответствующих подмножеств S_j.

3. Решение оптимизационных задач теории графов

Многие оптимизационные задачи теории графов допускают трактовку в терминах БЛП. Рассмотрим некоторые из таких задач.

3.1. Задача о максимальном паросочетании

Данная задача формулируется следующим образом: для заданного неориентированного графа требуется найти такое подмножество E^* рёбер из всего их множества E ($E^* \subset E$) максимальной мощности, в котором никакие два ребра не будут инцидентны (т.е. не будут иметь общей вершины).

В терминах БЛП формально данная задача формулируется следующим образом:

$$f = \sum_{j=1}^{n} x_i \to \max, \qquad (9)$$

$$\sum_{j=1}^{n} a_{ij} x_j \le 1,\ i = 1, \dots, m. \qquad (10)$$

Здесь x_j – переменные, ассоциированные с рёбрами, при этом $x_j = 1$, если ребро с номером j входит в максимальное паросочетание, и $x_j = 0$ в противоположном случае.

Коэффициенты a_{ij} – компоненты булевой матрицы инцидентности A размерности m×n – принимают значение 1, если вершина с номером i инцидентна ребру с номером j, и значение 0 – в противоположном случае.

Выражение (10) – ограничение в виде системы линейных неравенств – задаёт условие того, что в каждой вершине сумма инцидентных ей переменных (рёбер) не превышает единицы.

Задача о максимальном взвешенном паросочетании отличается только целевой функцией, т.е. вместо выражения (9) имеет место

$$f = \sum_{j=1}^{n} c_j x_j \to \max. \qquad (11)$$

Пусть, например, дан граф, изображённый на рис. 1.

В этом случае Matlab-код для решения данной задачи имеет вид:

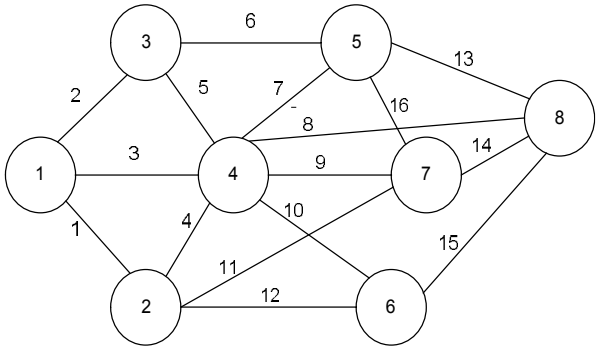

Рис. 1. Неориентированный граф

```
f = ones(16,1);
A = [1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0;
     1 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0;
     0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0;
     0 0 1 1 1 0 1 1 1 1 0 0 0 0 0 0;
     0 0 0 0 0 1 1 0 0 0 0 0 1 0 0 1;
     0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 0;
     0 0 0 0 0 0 0 0 1 0 1 0 0 1 0 1;
     0 0 0 0 0 0 0 1 0 0 0 0 1 1 1 0];
 b = ones(8,1);
 x = bintprog(-f, A, b).
```

В результате вычислений получится вектор значений булевых переменных:

```
x =
     0
     1
     0
     1
     0
     0
     0
     0
     0
     0
     0
     0
     0
     0
     1
     1.
```

Таким образом, $E^* = \{2,4,15,16\}$, т.е. в максимальное паросочетание входят рёбра с номерами 2, 4, 15, 16, что показано на рис. 2 (рёбра, входящие в максимальное паросочетание, выделены жирным).

3.2. Задача о минимальном вершинном покрытии

Данная задача формулируется следующим образом: для заданного неориентированного графа требуется найти такое подмножество V^* вершин из

всего их множества V ($V^* \subset V$) минимальной мощности, которые были бы инцидентны всем рёбрам графа.

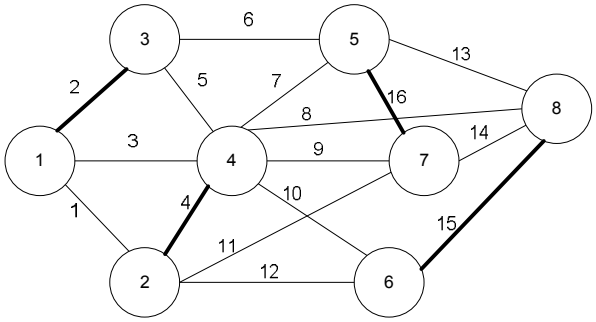

Рис. 2. Неориентированный граф

В терминах БЛП формально данная задача формулируется следующим образом:

$$f = \sum_{i=1}^{m} x_i \to \min, \qquad (12)$$

$$\sum_{j=1}^{m} a_{ij} x_j \geq 1, \; j = 1, \ldots, n. \qquad (13)$$

Коэффициенты a_{ij} – компоненты булевой матрицы инцидентности A размерности n×m – принимают значение 1, если ребро с номером j инцидентно вершине с номером i, и значение 0 – в противоположном случае.

Выражение (13) – ограничение в виде системы линейных неравенств – задаёт условие того, что каждому ребру инцидентна хотя бы одна вершина, т.е. сумма вершин, инцидентных каждому ребру, не меньше единицы.

Задача о минимальном *взвешенном* вершинном покрытии отличается только целевой функцией, т.е. вместо выражения (12) имеет место

$$f = \sum_{i=1}^{m} c_i x_i \to \min, \qquad (14)$$

где c_i – вес вершины с номером i.

Следует отметить, что данная задача является двойственной по отношению к задаче о максимальном паросочетании.

Решим, для примера, задачу о минимальном вершинном покрытии для графа, изображённого на рис. 1.

В этом случае Matlab-код для решения данной задачи имеет вид:

```
f = ones(8,1);
A = [1 1 0 0 0 0 0 0;
     1 0 1 0 0 0 0 0;
     1 0 0 1 0 0 0 0;
     0 1 0 1 0 0 0 0;
     0 0 1 1 0 0 0 0;
     0 0 1 0 1 0 0 0;
     0 0 0 1 1 0 0 0;
     0 0 0 1 0 0 0 1;
     0 0 0 1 0 0 1 0;
     0 0 0 1 0 1 0 0;
     0 1 0 0 0 0 1 0;
     0 1 0 0 0 1 0 0;
     0 0 0 0 1 0 0 1;
     0 0 0 0 0 0 1 1;
     0 0 0 0 1 0 1 0;
     0 0 0 0 1 0 1 0];
b = ones(16,1);
x = bintprog(f,-A,-b).
```

В результате вычислений получится вектор значений булевых переменных:

```
x =

     1
     0
     0
     1
     1
     1
     1
     0.
```

Таким образом, $V^* = \{1,4,5,6,7\}$, т.е. в минимальное покрытие входят вершины с номерами 1,4,5,6,7, что показано на рис. 3 (вершины, входящие в покрытие, выделены жирным).

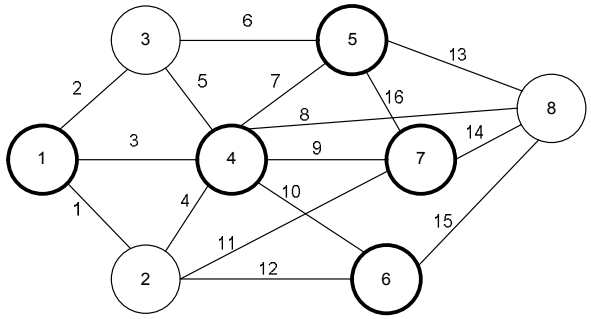

Рис. 3. Неориентированный граф

3.3. Задача о минимальном рёберном покрытии

Данная задача формулируется следующим образом: для заданного неориентированного графа требуется найти такое подмножество E^* рёбер из всего их множества E ($E^* \subset E$) минимальной мощности, что каждая вершина в графе будет инцидентна по крайней мере одному ребру из подмножества E^*.

В терминах БЛП формально данная задача формулируется следующим образом:

$$f = \sum_{j=1}^{n} x_j \to \min, \qquad (15)$$

$$\sum_{j=1}^{n} a_{ij} x_j \geq 1,\ i = 1, \ldots, m. \qquad (16)$$

Здесь x_j – переменные, ассоциированные с рёбрами, при этом $x_j = 1$, если ребро с номером j входит в минимальное покрытие, и $x_j = 0$ в противоположном случае.

Коэффициенты a_{ij} – компоненты булевой матрицы инцидентности A размерности m×n – принимают значение 1, если вершина с номером i инцидентна ребру с номером j, и значение 0 – в противоположном случае.

Выражение (16) – ограничение в виде системы линейных неравенств – задаёт условие того, что каждой вершине инцидентно хотя бы одно ребро, т.е. сумма рёбер, инцидентных каждой вершине, не меньше единицы.

Задача о минимальном *взвешенном* рёберном покрытии отличается только целевой функцией, т.е. вместо выражения (15) имеет место

$$f = \sum_{j=1}^{n} c_j x_j \to \min, \qquad (17)$$

где c_j – вес ребра с номером j.

Решим задачу о минимальном рёберном покрытии для графа, изображённого на рис. 1.

В этом случае Matlab-код для решения данной задачи имеет вид:

```
f = ones(16,1);
A = [1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0;
     1 0 0 1 0 0 0 0 0 0 1 1 0 0 0 0;
     0 1 0 0 1 1 0 0 0 0 0 0 0 0 0 0;
     0 0 1 1 1 0 1 1 1 1 0 0 0 0 0 0;
     0 0 0 0 0 1 1 0 0 0 0 0 1 0 0 1;
     0 0 0 0 0 0 0 0 0 1 0 1 0 0 1 0;
     0 0 0 0 0 0 0 0 1 0 1 0 0 1 0 1;
     0 0 0 0 0 0 0 1 0 0 0 0 1 1 1 0];
 b = ones(8,1);
 x = bintprog(f, -A, -b).
```

Результатом вычислений будет:

```
x =
  (1,1)      1
  (6,1)      1
 (10,1)      1
 (14,1)      1.
```

Таким образом, $E^* = \{1, 6, 10, 14\}$, т.е. в минимальное рёберное покрытие входят рёбра с номерами 1, 6, 10, 14, что показано на рис. 4 (рёбра, входящие в покрытие, выделены жирным).

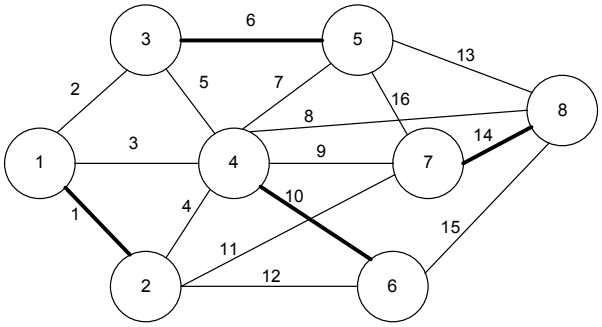

Рис. 4. Неориентированный граф

3.4. Задача о максимальном независимом множестве вершин

Данная задача формулируется следующим образом: для заданного неориентированного графа из всего множества вершин V требуется найти максимальное по величине подмножество V^* ($V^* \subset V$) попарно несмежных вершин.

В терминах БЛП формально данная задача формулируется следующим образом:

$$f = \sum_{i=1}^{m} x_i \to \max, \qquad (18)$$

$$\sum_{i=1}^{m} a_{ij} x_j \leq 1,\ j = 1, \ldots, n. \qquad (19)$$

Здесь x_i – переменные, ассоциированные с вершинами, при этом $x_i = 1$, если вершина с номером i входит в максимальное независимое множество вершин, и $x_i = 0$ в противоположном случае.

Коэффициенты a_{ij} – компоненты булевой матрицы инцидентности A размерности m×n – принимают значение 1, если ребро с номером j инцидентно вершине с номером i, и значение 0 – в противоположном случае.

Выражение (19) – ограничение в виде системы линейных неравенств – задаёт условие того, что каждому ребру сумма инцидентных ему переменных (вершин) не превышает единицы.

Задача о максимальном *взвешенном* независимом множестве вершин отличается только целевой функцией, т.е. вместо выражения (18) имеет место

$$f = \sum_{i=1}^{m} c_i x_i \to \max. \qquad (20)$$

Решим задачу о максимальном независимом множестве вершин для графа, изображённого на рис. 1.

В этом случае Matlab-код для решения данной задачи имеет вид:

```
f = ones(8,1);
A = [1 1 0 0 0 0 0 0;
     1 0 1 0 0 0 0 0;
     1 0 0 1 0 0 0 0;
     0 1 0 1 0 0 0 0;
     0 0 1 1 0 0 0 0;
     0 0 1 0 1 0 0 0;
     0 0 0 1 1 0 0 0;
     0 0 0 1 0 0 0 1;
     0 0 0 1 0 0 1 0;
     0 0 0 1 0 1 0 0;
     0 1 0 0 0 0 1 0;
     0 1 0 0 0 1 0 0;
     0 0 0 0 1 0 0 1;
     0 0 0 0 0 0 1 1;
     0 0 0 0 0 1 0 1;
     0 0 0 0 1 0 1 0];
 b = ones(16,1);
 x = bintprog(-f,A,b).
```

В результате вычислений получится вектор значений булевых переменных:

```
x =
    1
    0
    0
    0
    0
    1
    1
    0.
```

Таким образом, $V^* = \{1,6,7\}$, т.е. в максимальное независимое множество вершин входят вершины с номерами 1,6,7, что показано на рис. 5 (вершины, входящие в максимальное независимое множество, выделены жирным).

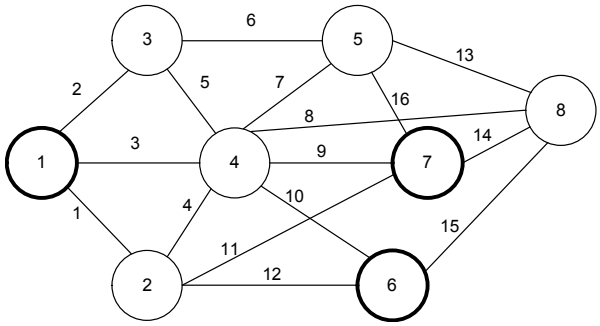

Рис. 5. Неориентированный граф

Выводы

Несмотря на отсутствие в системе Matlab встроенных функций для решения задач комбинаторной оптимизации, некоторые из таких задач, а именно – задачи, допускающие интерпретацию в виде задач булевого линейного программирования, в том числе некоторые оптимизационные задачи теории графов (задача о максимальном паросочетании, задача о минимальном вершинном покрытии, задача о минимальном рёберном покрытии, задача о максимальном независимом множестве вершин и др.), могут быть решены с помощью встроенной функции bintprog. Для таких задач, как задача о назначении, задача коммивояжёра, задача о рюкзаке, задача о покрытии множества, разработаны функции, которые могут быть полезны пользователям системы Matlab.

Литература

1. Иглин, С.П. Решение некоторых задач теории графов в MATLAB [Текст] / С.П. Иглин // Exponenta Pro. Математика в приложениях. – 2004. – №4(4). – С. 28–33.

2. Иглин, С.П. Математические расчёты на базе MATLAB [Текст] / С.П. Иглин. - СПб.: БХВ-Петербург, 2005. – 640 с.

3. Matlog: Logistics Engineering MATLAB Toolbox [Электронный ресурс]. – Режим доступа: http://www.ise. ncsu.edu/kay/matlog. – 12.03.2013.

4.TOMLAB Optimization Inc. [Электронный ресурс]. – Режим доступа: http://tomopt.com/tomlab/optimization/miqp.php. – 12.03.2013.

5. Зайченко, Ю.П. Исследование операций [Текст] / Ю.П. Зайченко. – 3-е изд., перераб. и доп. – К.: Выща шк. Головное изд-во, 1988. – 552 с.

Поступила в редакцию 04.06.2013, рассмотрена на редколлегии 12.06.2013

Рецензент: д-р техн. наук, проф., заведующий кафедрой компьютерных систем и сетей В.С. Харченко, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков, Украина.

ПРО РІШЕННЯ ЗАДАЧ ДИСКРЕТНОЇ ОПТИМІЗАЦІЇ В СИСТЕМІ КОМП'ЮТЕРНОЇ МАТЕМАТИКИ MATLAB

І.В. Лисенко

Розглянуто можливості рішення задач дискретної оптимізації в системі комп'ютерної математики Matlab. Надано опис розроблених в середовищі Matlab вбудованих функцій для рішення деяких задач комбінаторної оптимізації (0-1-задачі про рюкзак, задачі про покриття множини, задачі комівояжера, задачі про призначення), а також наводяться приклади рішення оптимізаційних задач теорії графів, що допускають інтерпретацію у вигляді задач булевого лінійного програмування, до числа яких відносяться: задача про максимальне паросполучення, задача про мінімальне вершинне покриття, задача про мінімальне реберне покриття, задача про максимальну незалежну множину вершин.

Ключові слова: дискретна оптимізація, задачі булевого лінійного програмування.

ABOUT SOLVING OF DISCRETE OPTIMIZATION PROBLEMS IN SYSTEM OF COMPUTER MATHEMATICS MATLAB

I.V. Lysenko

Possibilities of solving of the discrete optimization problems in program system Matlab are considered. Created in Matlab embedded functions for solving of some combinatorial optimization problems (0-1 knapsack problem, set covering problem, travelling salesman problem, assignment problem) are described. An examples of solving of graph theory optimization problems, that can be presented as a Boolean linear programming problems, are given. These tasks include: the problem of the maximal matching, the problem of the minimal vertex cover, the problem of the minimal edge covering, the problem of maximal independent set of vertices.

Key words: discrete optimization, Boolean linear programming problems.

Лысенко Игорь Владимирович – канд. техн. наук, доцент, доцент кафедры компьютерных систем и сетей, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков, Украина, e-mail: I.Lysenko@csn.khai.edu.