Журнал обчислювальної та прикладної математики **2014, №3(117)** Journal of Computational & Applied Mathematics

УДК 519.85

О РЕШЕНИИ МИНИМАКСНЫХ ЗАДАЧ РАЗМЕЩЕНИЯ ИСТОЧНИКОВ ФИЗИЧЕСКОГО ПОЛЯ

П. И. Стецюк, В. И. Ляшко, С. И. Яремчук

Резюме. Исследуется эффективность программы gurobi для решения задачи линейного программирования с булевыми переменными, соответствующей минимаксной задаче размещения источников физического поля. Определяются количества точек измерения поля и мест размещения источников, при которых можно найти решение задачи за несколько часов. Для релаксированной задачи оценивается время решения с помощью gurobi, а также построены две равносильные задачи негладкой оптимизации.
Ключевые слова: минимаксная задача размещения, ЛП-задача с булевыми переменными, релаксированная задача (ЛП-задача), gurobi.

Введение

При проектировании систем, качество работы которых зависит от физических полей, генерируемых компонентами системы (поля могут быть тепловыми, диффузионными и др.), возникает следующая задача. В заданной области требуется разместить источники полей на фиксированные места так, чтобы максимальное из значений поля в заданных точках области было наименьшим. Если соответствующее физическое поле описывается линейной краевой задачей, то математическая модель этой задачи оптимизации может быть представлена таким образом[1, 2]:

$$f(x) = \max_{k\in[1:K]} f_k(x) \to \min, \tag{1}$$

где

$$f_k(x) = \sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}^k x_{ij}, \quad k \in [1:K], \tag{2}$$

при ограничениях

$$\sum_{i=1}^{N} x_{ij} = 1, \quad j \in [1:N], \tag{3}$$

$$\sum_{j=1}^{N} x_{ij} = 1, \quad i \in [1:N]. \tag{4}$$

Здесь булевая переменная

$$x_{ij} = \begin{cases} 1, & \text{если } i\text{-й источник размещается на } j\text{-ом месте,} \\ 0, & \text{если } i\text{-й источник не размещается на } j\text{-ом месте,} \end{cases} \tag{5}$$

$i, j \in [1 : N]$, где N — количество мест размещения источников; K — количество точек измерения значения поля; c_{ij}^k — вклад i-го источника при назначении его на j-ое место ($i \in [1 : N]$, $j \in [1 : N]$) в значение поля в k-ой точке измерения ($k \in [1 : K]$).

Предполагается, что K — небольшое по сравнению с N. Если $N > 60$, то задача (1)–(5) является задачей больших размеров [1]. Цель статьи — найти такие параметры N и K, при которых за несколько часов можно получить решение задачи (1)–(5) с помощью программы gurobi [3], которая считается одной из наилучших для решения задач линейного и целочисленного программирования. Задача (1)–(5) сводится к задаче линейного программирования (ЛП-задача) с булевыми переменными. Для генерации c_{ii}^k используются равномерно распределенные на интервале $[-100, 100]$ случайные числа, которые содержат две значащих десятичных цифры после точки. Такая модель рассчитана на наихудший случай представления физического поля, где соседние компоненты c_{ij}^k для одного и того же k могут сильно отличаться друг от друга.

Последовательность изложения материала следующая. В разделе 1 приведена ЛП-задача с булевыми переменными и описаны два тестовых эксперимента. Первый эксперимент связан с оценкой N при $K = 20$, а второй эксперимент — с оценкой K при $N = 100$. В разделе 2 описан тестовый эксперимент для оценки времени решения нецелочисленной ЛП-задачи при $N = 100$ и $N = 200$, что позволяет определить потенциальные границы использования симплекс-метода и метода внутренних точек в схемах ветвей и границ для решения ЛП-задачи с булевыми переменными. Для всех трех тестовых экспериментов даны коды на языке моделирования AMPL [4]. В третьем разделе приведены две задачи негладкой оптимизации для нахождения оптимального значения целевой функции в ЛП-задаче.

1. ЛП-задача с булевыми переменными и программа gurobi

Задачу (1)–(5) можно сформулировать в форме ЛП-задачи с булевыми переменными [1]. Она имеет следующий вид:

$$z \to \min \tag{6}$$

при ограничениях

$$\sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij}^k x_{ij} \le z, \quad k = [1 : K], \tag{7}$$

$$\sum_{i=1}^{N} x_{ij} = 1, \quad j \in [1 : N], \tag{8}$$

$$\sum_{j=1}^{N} x_{ij} = 1, \quad i \in [1:N], \tag{9}$$

$$x_{ij} \in \{0,1\}, \quad i \in [1:N],\ j \in [1:N]. \tag{10}$$

Для решения задачи (6)–(10) можно использовать программу gurobi. Ниже дан код на языке AMPL, где gurobi установлена в качестве солвера для двух задач: BLP — задача (6)–(10) и LP — релаксированная BLP, полученная ослаблением булевых переменных в задаче (6)–(10). Последняя будет рассмотрена в разделе 2.

```
# section 1: parameters and variables
param Table1{i in 1..10, j in 1..5};
param N := 17; param K:=20; param irun default 0;
param c{k in 1..K, i in 1..N, j in 1..N};
var x {i in 1..N, j in 1..N} binary; var z;
var y {i in 1..N, j in 1..N} >=0;
# section 2: objective and constraints
minimize obj: z;
subject to con7x {k in 1..K}:
   sum {i in 1..N, j in 1..N} c[k,i,j]*x[i,j] <= z;
subject to con8x {j in 1..N}: sum {i in 1..N} x[i,j] = 1;
subject to con9x {i in 1..N}: sum {j in 1..N} x[i,j] = 1;
subject to con7y {k in 1..K}:
   sum {i in 1..N, j in 1..N} c[k,i,j]*y[i,j] <= z;
subject to con8y {j in 1..N}: sum {i in 1..N} y[i,j] = 1;
subject to con9y {i in 1..N}: sum {j in 1..N} y[i,j] = 1;
# section 3: solver, BLP and LP problem
option solver gurobi;
problem BLP: obj,x,z,con7x,con8x,con9x;
problem LP: obj,y,z,con7y,con8y,con9y;
# section 4: 10 tests and Table1
repeat {
  let irun:=irun+1;
  for {k in 1..K, i in 1..N, j in 1..N}{
     let c[k,i,j] := round(Uniform(-1,1)*10000.)/100.;
  }
  solve BLP; display z, _solve_time;
  let Table1[irun,1] := obj; let Table1[irun,2] := _solve_time;
  solve LP; display z, _solve_time;
  let Table1[irun,3] := obj; let Table1[irun,4] := _solve_time;
  let Table1[irun,5] := Table1[irun,1]-Table1[irun,3];
  let Table1[irun,5] := Table1[irun,5]/max(1,abs(Table1[irun,3]));
} while irun <= 9;
display K, N, Table1;
```

AMPL-код оформлен в виде 4-х секций. Первая секция содержит описание параметров и переменных обеих задач. Вторая секция описывает

целевую функцию и ограничения задач. Третья секция определяет солвер, формирует BLP и LP задачи. Четвертая секция генерирует величины c_{ij}^k, осуществляет вычисления для десяти тестовых примеров и готовит выходную таблицу Table1.

Значения c_{ij}^k для $i, j \in N$ и $k \in K$ генерируется с помощью AMPL-функций *Uniform* и *round*. Значение c_{ij}^k равно *round(Uniform(-1,1)*10000.)/100.*, т. е. случайному числу в интервале $[-100, 100]$ с точностью до двух значащих десятичных цифр после точки. Цикл по индексам i, j и k построен таким образом, чтобы обеспечить равномерно распределенные случайные величины для каждой k-ой точки измерения поля, $k \in K$.

Таблица Table1 формируется следующим образом. Ее строки соответствуют десяти различным расчетам. Столбцы 1 и 2 содержат оптимальное значение целевой функции и полное время решения (в секундах) для задачи BLP. Столбцы 3 и 4 содержат такие же величины, но для задачи LP. Столбец 5 содержит относительный разрыв между оптимальными значениями целевых функций BLP и LP задач. Он вычисляется как отношение их разности к модулю оптимального значения целевой функции задачи LP.

Результаты работы этого кода под студенческой версией AMPL, которая включает программу gurobi версии 2.0.1, представлены в таблице:

```
Gurobi 2.0.1: K = 20, N = 17, Table1 [*,*]
        1         2          3          4          5
1    -132.69    509.609   -245.13    0.015625   0.458696
2    -152.11   1306.53    -274.662   0.015625   0.446193
3    -176.66    538.016   -289.772   0.015625   0.390349
4    -181.92    214.875   -284.107   0.015625   0.359678
5    -199.31    529.438   -313.041   0.015625   0.36331
6    -188.68    465.891   -291.875   0.015625   0.353558
7    -190.57   1671.81    -302.283   0          0.369564
8    -201.22    158.516   -307.906   0.015625   0.346488
9    -197.96    230.672   -310.64    0.015625   0.362736
10   -203.97    279.328   -314.035   0.015625   0.350486
```

Здесь $K = 20$ и $N = 17$ выбраны из следующих соображений. Если $N = 17$, то BLP и LP задачи содержат 290 переменных, из них $289 = 17 \times 17$ переменных x_{ij} и одна переменная z. Студенческая версия AMPL поддерживает решение задач математического программирования до 300 переменных. $K = 20$ выбрано из соображений приемлимого времени для решения BLP задачи. Вычисления проводились на компьютере Pentium E5200/2.5GHz/4 GB в среде Windows Server.

Из таблицы видим, что время решения тестовых BLP задач составляет от чуть меньше трех минут (восьмая задача) до немногим более 30 минут (седьмая задача). При этом LP задачи решаются почти мгновенно, т. е. за сотые доли секунды. Это обьясняется очень большим количеством симплексных итераций и обусловлено тем, что относительный разрыв между целевыми функциями BLP и LP задач составляет не менее тридцати процентов, а для первой и второй задач — больше 40 процентов. Так, например,

для второй BLP задачи программа Gurobi 2.0.1 нашла оптимальное решение (objective -152.11) за 16.881.239 симплексных итераций при 1.699.284 вершинах ветвления. При этом оптимальное решение LP задачи (objective -274.662447) было найдено всего за 160 симплексных итераций.

Второй эксперимент для программы gurobi состоял в проверке того, для каких K можно решить BLP задачу при $N = 100$ за приемлемое время. Ему соответствует такой код на языке AMPL.

```
# section 1: parameters and variables
param Table2{i in 1..10, j in 1..7};
param N := 100; param K default 1; param irun default 0;
param c{k in 1..K, i in 1..N, j in 1..N};
var x {i in 1..N, j in 1..N} binary; var z;  #
var y {i in 1..N, j in 1..N} >=0;
#  section 2: objective and constraints
minimize obj: z;
subject to con7x {k in 1..K}:
   sum {i in 1..N, j in 1..N} c[k,i,j]*x[i,j] <= z;
subject to con8x {j in 1..N}: sum {i in 1..N} x[i,j] = 1;
subject to con9x {i in 1..N}: sum {j in 1..N} x[i,j] = 1;
subject to con7y {k in 1..K}:
   sum {i in 1..N, j in 1..N} c[k,i,j]*y[i,j] <= z;
subject to con8y {j in 1..N}: sum {i in 1..N} y[i,j] = 1;
subject to con9y {i in 1..N}: sum {j in 1..N} y[i,j] = 1;
#  section 3: set BLP and LP problems
problem BLP: obj,x,z,con7x,con8x,con9x;
problem LP: obj,y,z,con7y,con8y,con9y;
#  section 4: 10 tests and Table2
for {ik in 5..6, ir in 1..5} {
   let irun:=irun+1; let K:=ik;
   let Table2[irun,1] := K; let Table2[irun,2] := ir;
   for {k in 1..K, i in 1..N, j in 1..N}{
      let c[k,i,j] := round(Uniform(-1,1)*10000.)/100.;
   }
   solve BLP; display z, _solve_time;
   let Table2[irun,3] := obj; let Table2[irun,4] := _solve_time;
   solve LP; display z, _solve_time;
   let Table2[irun,5] := obj; let Table2[irun,6] := _solve_time;
   let Table2[irun,7] := Table2[irun,3]-Table2[irun,5];
   let Table2[irun,7] := Table2[irun,7]/max(1,abs(Table2[irun,5]));
}
display N, Table2;
```

Он оформлен по аналогии с предыдущим в виде 4-х секций. Первая и вторая секции содержат описание параметров и переменных, целевой функции

и ограничений. Третья секция определяет BLP и LP задачи, но не устанавливает тип солвера. Это означает, что код может быть использован и другими программами NEOS-солвера [5], которые поддерживают решение задач смешанного линейно-целочисленного программирования. Для них нет ограничений на количество переменных, как в студенческой версии AMPL.

Четвертая секция генерирует величины c_{ij}^k по тому же правилу, как выше, но осуществляет расчеты для пяти тестовых примеров при $K = 5$ и пяти тестовых примеров при $K = 6$. Здесь же заполняется выходная таблица Table2, аналогичная по данным Table1, но с тем отличием, что в столбцах 1 и 2 располагаются значения K и соответствующие им номера тестовых примеров. Столбцы 3–7 в таблице Table2 заполняются аналогично столбцам 1–5 таблицы Table1.

Результатом работы программы gurobi версии 5.5.0 из NEOS-солвера есть следующая таблица.

```
Gurobi 5.5.0: N = 100, Table2 [*,*]
     1   2      3          4          5          6          7
1    5   1   -5869.42    400.626   -5892.05   1.44978   0.00384047
2    5   2   -5710.5     282.692   -5734.12   0.26196   0.00412004
3    5   3   -5871.45    349.114   -5891.31   0.498923  0.00337047
4    5   4   -5767.9     409.281   -5790.15   0.384941  0.00384262
5    5   5   -5706.69    458.018   -5727.5    0.619906  0.00363362
6    6   1   -5285.61   4329.85    -5318.75   1.43478   0.00623152
7    6   2   -5260.29   5996.05    -5289.63   0.643902  0.00554717
8    6   3   -5272.42   1585.42    -5301.35   0.965853  0.00545705
9    6   4   -5334.62   2723.5     -5366.18   0.893864  0.00588089
10   6   5   -5345.87   3186.58    -5373.33   0.527919  0.00511129
```

Из нее видим, что, если $K = 6$, то время решения BLP задач составляет от 1585 до 5996 секунд, т.е максимальное время решения не превышает двух часов. При $K = 7$ решить BLP задачу за несколько часов не представляется возможным. Это легко видеть из того, что время решения при $K = 6$ приблизительно в десять раз больше, чем время решения при $K = 5$.

Следовательно, второй эксперимент показывает, что, если $N = 100$, то решать задачи (1)–(5) для наихудшего случая физического поля можно, если использовать до 6 точек измерений поля. Реальные физические поля могут поднять эту границу, но это требует дополнительного исследования.

2. ЛП-задача и программа gurobi

Если для решения задачи (6)–(10) применяется метод ветвей и границ, то для нахождения оценок требуется решать ЛП-задачу, которая получена релаксацией (ослаблением) булевых ограничений (10). ЛП-задача имеет такой вид:

$$z \to \min \tag{11}$$

при ограничениях

$$\sum_{i=1}^{N}\sum_{j=1}^{N} c_{ij}^{k} x_{ij} \le z, \quad k \in [1:K], \tag{12}$$

$$\sum_{i=1}^{N} x_{ij} = 1, \quad j \in [1:N], \tag{13}$$

$$\sum_{j=1}^{N} x_{ij} = 1, \quad i \in [1:N], \tag{14}$$

$$x_{ij} \ge 0, \quad i \in [1:N],\ j \in [1:N]. \tag{15}$$

Здесь верхние границы на переменные x_{ij} отсутствуют, так как они следуют из ограничений (13) и (14) при условии (15).

Для решения ЛП-задачи (11)–(15) могут быть использованы стандартные программы для задач линейного программирования, в том числе и программа gurobi. Поэтому третий эксперимент состоял в оценке времени решения ЛП-задач (11)–(15) для $N = 100$ и $N = 200$ с помощью gurobi. Рассматривались два времени — общее время решения и время работы процессора. Последнее является более важным, так как оно адекватнее отражает многократный вызов ЛП-программы.

Третий эксперимент реализует следующий AMPL-код:

```
# section 1: parameters and variables
param Table3{i in 1..10, j in 1..6};
param N default 100;  param K := 10;
param c{k in 1..K, i in 1..N, j in 1..N};
var x {i in 1..N, j in 1..N} >=0; var z;
#  section 2: objective and constraints
minimize obj: z;
subject to con11x {k in 1..K}:
   sum {i in 1..N, j in 1..N} c[k,i,j]*x[i,j] <= z;
subject to con12 {j in 1..N}: sum {i in 1..N} x[i,j] = 1;
subject to con13 {i in 1..N}: sum {j in 1..N} x[i,j] = 1;
#  section 3: 20 tests and Table3
for {ip in 1..2, irun in 1..10} {
   let N := 100*ip;
   for {k in 1..K, i in 1..N, j in 1..N}{
      let c[k,i,j] := round(Uniform(-1,1)*10000.)/100.;
   }
   solve; display z, _solve_time,_solve_system_time;
   let Table3[irun,3*(ip-1)+1]:=obj;
   let Table3[irun,3*(ip-1)+2] := _solve_time;
   let Table3[irun,3*(ip-1)+3] := _solve_system_time;
}
display K, Table3;
```

Он состоит из трех секций. В первой секции описаны параметры и переменные, во второй — целевая функция и ограничения ЛП-задачи. В третьей секции выполняются двадцать тестовых расчетов для случайно сгенерированных c_{ij}^k: из них десять — для задачи LP100 ($N = 100$) и десять — для задачи LP200 ($N = 200$). Параметр K выбирался равным 10 (промежуточный между наименьшим $K = 6$ и наибольшим $K = 20$ из предыдущих экспериментов).

Результатом третьего эксперимента на NEOS-солвере является следующая таблица:

Gurobi 5.5.0: K = 10, Table3 [*,*]

	1	2	3	4	5	6
1	-4203.85	0.734888	0.104984	-9350.55	3.2845	0.407938
2	-4248.12	0.392939	0.008998	-9150.88	3.14152	0.019997
3	-4150.84	0.402939	0.007999	-9175.7	2.79957	0.023996
4	-4143.26	0.356946	0.004999	-9213.25	3.80342	0.017997
5	-4126.99	0.320951	0.005999	-9157.11	3.39848	0.019997
6	-4288.25	0.338949	0.007999	-9113.78	3.54446	0.023997
7	-4284.13	0.391941	0.009999	-9235.04	2.77158	0.020996
8	-4215.55	0.310952	0.007998	-9259.78	2.84757	0.017998
9	-4181.55	0.353946	0.005999	-9218.42	3.02554	0.013997
10	-4078.52	0.338949	0.005	-9300.16	2.96055	0.019997

Ее строки соответствуют десяти различным тестам для задач LP100 и LP200. Первые 3 столбца соответствуют задаче LP100, а вторые три столбца — задаче LP200. Столбцы 1 и 4 содержат значения целевых функций, столбцы 2 и 5 — полное время работы программы gurobi, а столбцы 3 и 6 — время работы процессора.

Из таблицы Table3 видим, что время работы процессора для первой серии ЛП-задач составляет порядка 0.01 секунды, а для второй серии ЛП-задач — порядка 0.04 секунды. Первый тест здесь не следует принимать во внимание, так как он включает также затраты и самого AMPL на подготовку моделей LP100 и LP200, которые значительно превосходят затраты на решение ЛП-задач. Эти времена работы процессора можно использовать для грубых оценок затрат на решение ЛП-задач в схемах метода ветвей и границ, если $N = 100$ либо $N = 200$. В действительности, использование gurobi будет эффективнее, так как оптимальные решения ЛП-задач в ряде вершин ветвления незначительно отличаются друг от друга.

Программу gurobi можно использовать для нахождения приближенного решения ЛП-задачи с булевыми переменными посредством последовательного округления переменных к единицам (означает прикрепление источников к местам их размещения), решая последовательно ЛП-задачи, количество которых не превосходит N — количества источников. Это легко реализовать на том же языке AMPL и время работы такой программы на NEOS-солвере не будет превышать 10 минут, если $N = 200$.

3. Две лагранжевые оценки для ЛП-задачи

Для решения задачи (11)–(15) можно применять методы негладкой оптимизиции, используя лагранжеву релаксацию по части ограничений [6]. Это позволяет учесть специфические особенности матрицы ограничений задачи (11)–(15), а именно то, что ограничения (13)–(14) являются ограничениями классической задачи о назначениях [6].

Сформулируем две задачи негладкой оптимизации для нахождения оптимального значения целевой функции в задаче (11)–(15), которое для удобства дальнейшего изложения обозначим z^*. Из теории двойственности следует справедливость приведенных ниже утверждений.

Лемма 1. *Пусть z^* — минимальное значение целевой функции в задаче (11)–(15). Тогда $z^* = \psi_1^*$, где ψ_1^* является решением задачи негладкой оптимизации*

$$\psi_1^* = \psi_1(u^*) = \max_{u \geq 0} \psi_1(u), \tag{16}$$

где вогнутая негладкая функция $\psi_1(u)$ от переменных $u = (u_1, \ldots, u_K)$ определена следующим образом:

$$\psi_1(u) = \min \sum_{i=1}^{N} \sum_{j=1}^{N} \left(\sum_{k=1}^{K} u_k c_{ij}^k \right) x_{ij} + (1 - \sum_{k=1}^{K} u_k) z, \tag{17}$$

$$\sum_{i=1}^{N} x_{ij} = 1, \quad j \in [1:N], \tag{18}$$

$$\sum_{j=1}^{N} x_{ij} = 1, \quad i \in [1:N], \tag{19}$$

$$x_{ij} \geq 0, \ i, j \in [1:N], \quad N \min_{i,j,k} \{c_{ij}^k\} \leq z \leq N \max_{i,j,k} \{c_{ij}^k\}. \tag{20}$$

Для решения задачи (16) можно использовать методы минимизации негладких выпуклых функций [6]. Количество переменных в ней равно K. Вычислительная сложность алгоритма определяется выбранными методом негладкой оптимизации и методом решения линейной задачи о назначениях с $N \times N$ переменными x_{ij}. Решение последней используется для вычисления значения функции $\psi_1(u)$ и ее суперградиента.

Задача (16) получена лагранжевой релаксацией ЛП-задачи (11)–(15) по ограничениям (12), которым и соответствуют неотрицательные множители Лагранжа $u_1, \ldots, u_K$. Если же использовать лагранжеву релаксацию по ограничениям (12) и (13) одновременно, то получаем такое утверждение.

Лемма 2. *Минимальное значение $z^* = \psi_2^*$, где ψ_2 можно получить как решение задачи негладкой оптимизации*

$$\psi_2^* = \psi_2(u^*, v^*) = \max_{u \geq 0} \psi_2(u, v), \tag{21}$$

где вогнутая негладкая функция $\psi_2(u,v)$ *от переменных* $u=(u_1,\ldots,u_K)$ *и* $v=(v_1,\ldots,v_N)$ *определена следующим образом:*

$$\psi_2(u)=\min\sum_{i=1}^{N}\sum_{j=1}^{N}\left(v_j+\left(\sum_{k=1}^{K}u_k c_{ij}^k\right)\right)x_{ij}+(1-\sum_{k=1}^{K}u_k)z-\sum_{j=1}^{N}v_j, \quad (22)$$

$$\sum_{j=1}^{N}x_{ij}=1, \quad i\in[1:N], \quad (23)$$

$$x_{ij}\geq 0,\ i,j\in[1:N], \quad N\min_{i,j,k}\{c_{ij}^k\}\leq z\leq N\max_{i,j,k}\{c_{ij}^k\}. \quad (24)$$

В задаче (21) количество переменных равно $K+N$. Поэтому трудоемкость итерации метода негладкой оптимизации будет выше, чем при решении задачи (16). Но зато более простой является подзадача (22)–(24), решение которой требуется для вычисления значения функции $\psi_2(u,v)$ и ее суперградиента. Чтобы ее решить требуется N раз найти минимальный элемент в одномерном массиве длины N. Это значительно проще, чем решать линейную задачу о назначениях.

Алгоритмы на основе лемм 1 и 2 могут быть сориентированы на случай больших N (порядка тысяч) и сравнительно небольших K (до 20). Такие размеры трудны для ЛП-программ общего назначения, так как при $N=1000$ будем иметь K миллионов величин c_{ij}^k и только для их хранения потребуются мегабайты оперативной памяти. Поэтому субградиентные алгоритмы для решения задач (16) и (21) являются перспективными, так как они способны значительно поднять границы для эффективно решаемых задач вида (1)–(5) по отношению к тому, что позволяет либо симплекс-метод, либо метод внутренних точек. При использовании субградиентных методов легко учесть специфику ЛП-задачи (11)–(15), а также уточнять верхнюю оценку для целевой функции в ЛП-задаче с булевыми переменными, используя полученные на итерации множители Лагранжа.

Заключение

Для минимаксной задачи размещения источников физического поля результаты численных экспериментов с программой gurobi показали, что чем больше разрыв двойственности в ЛП-задаче с булевыми переменными, то тем меньше должно быть количество точек измерения значения поля и мест размещения источников, чтобы задача могла быть решена за приемлемое время. Однако, эти расчеты проводились на модельном примере, который генерировался датчиком случайных чисел из расчета на наихудший случай. Интересным представляется изучение возможностей решения подобных задач не на модельном примере, а, например, для реальной задачи о радиационной печи, для которой расчет температурного поля в заданной области требует решения краевой задачи для уравнения теплопроводности.

Подготовка модели на языке AMPL, с последующим расчетом с помощью gurobi на NEOS-солвере, позволяет быстро и с небольшими затратами

проверить возможности решения реальных задач. Для требуемых размеров, если разрыв двойственности малый, то программа gurobi может успешно находить точные решения ЛП-задач с булевыми переменными. Если же разрыв двойственности большой, то на основе gurobi и AMPL легко строить приближенные алгоритмы, которые могут находить вполне приемлимые для практики решения реальных задач.

Для построения точных и приближенных алгоритмов перспективным представляется путь с использованием методов негладкой оптимизации, который позволяет учесть специфические особенности матрицы ограничений и увеличить размеры решаемых задач. В сочетании с применением быстро сходящихся вариантов субградиентных методов с преобразованием пространства [7] он способен обеспечить эффективные алгоритмы для решения оценочных ЛП-задач. Простые матрично-векторные операции делают эти алгоритмы перспективными для реализации в системах параллельных или распределенных вычислений. Существующие библиотеки стандартных программ для параллельных матрично-векторных операций позволяют в краткие сроки адаптировать такие алгоритмы для векторных процессоров на основе графических ускорителей (GPU) и других.

Литература

1. Яремчук С. І., Іващенко С. М. Побудова методу гілок та меж для розв'язання мімаксної задачі про призначення // Тези VII Міжнародної науково-технічної конференції "Інформаційно-компьютерні технології 2014" (29–30 травня 2014 року). — Житомир: ЖДТУ, 2014. — С. 61–64.
2. Яремчук С. І., Бурда Р. В. Р-алгоритм розв'язання дискретної мінімаксної задачі розміщення джерел фізичного поля // Журнал обчислювальної та прикладної математики. — 2009. — № 2. — С. 119–133.
3. Gurobi Optimization, Inc., Gurobi Optimizer Reference Manual, 2014, `http://www.gurobi.com`.
4. Fourer R. AMPL, A Modeling Language for Mathematical Programming, Second Edition / R. Fourer, D. Gay, B. Kernighan. — Belmont: Duxburry Press, 2003. — 517 p.
5. NEOS Solver [Электронный ресурс]: `http://www.neos-server.org/neos/solvers/`. — Режим доступа: свободный.
6. Михалевич В. С., Трубин В. А., Шор Н. З. Оптимизационные задачи производственно-транспортного планирования. — М.: Наука, 1986. — 264 с.
7. Стецюк П. И. Методы эллипсоидов и r-алгоритмы. — Кишинэу: Эврика, 2014. — 488 с.

Институт кибернетики им. В.М.Глушкова НАН Украины, пр. Академика Глушкова, 40, Киев-187, 03680, Украина.
Национальный университет "Киево-Могилянская академия", ул. Сковороды,2, Киев, 04070, Украина.
Житомирский государственный технологический университет, ул. Черняховского, 103, Житомир, 10005, Украина.

Поступила 27.06.2014