УДК 004.438:519.85

А.Л. Гольдштейн

Пермский национальный исследовательский политехнический университет,
Пермь, Россия

МОДЕЛИРОВАНИЕ В LINGO

В статье рассмотрены основы современного языка моделирования, используемого в пакете оптимизации LINGO и фактически не освещенного в отечественных публикациях. Дана концепция языка, базирующаяся на описании задачи множествами подобных объектов, обладающих общими атрибутами, и функциях, оперирующих с этими множествами. Конструкция языка учитывает структурированность большинства задач математического программирования и их многообразие. Набор функций позволяет моделировать практически любые задачи математического программирования: прогнозирования, планирования, конструирования, организации производства, размещения, логистики, эффективной реализации бизнес-процессов, экономические и финансовые проблемы в ситуациях определенности, риска и неопределенности и т.д. Показаны особенности и преимущества языка при моделировании и решении задач математического программирования средней и большой размерности, в частности, компактность представления математической модели и легкость масштабирования моделируемых задач. Структурированное представление модели задачи позволяет отделить модель оптимизации от данных, выделить разделы предварительных вычислений (например, средних и дисперсии) и задания начальных условий. Рассматриваемый в статье язык моделирования дает возможность в одном описании задать вариантность решения задачи, выделить подмодели и из них формировать необходимые комбинации, каждая из которых решается как единая задача. Язык включает средства программирования, которые позволяют получать данные как заданные непосредственно в описании, так и из внешних источников (текстовых файлов, электронных таблиц, баз данных и пользовательских приложений), а также передавать результаты решения на экран или во внешние файлы, базы данных и т.д. Приведенные в статье примеры демонстрируют гибкость языка моделирования LINGO и компактность моделей, получаемых с его помощью.

Ключевые слова: язык моделирования, задачи математического программирования, LINGO.

A.L. Goldshtein

Perm National Research Polytechnic University, Perm, Russian Federation

MODELING WITH LINGO

The article describes the foundations of the modern modeling language used in the package of optimization LINGO, and in fact, not illuminated in national publications. The concept of language is based on the description of the task by sets of similar objects with common attributes, and on the functions that operate with these sets. The design of language takes into account the structure of most mathematical programming problems and their variety. Set of functions allows you to simulate practically any tasks of mathematical programming: forecasting, planning, design, organization of production, distribution, logistics, the effective implementation of business processes, economic and financial problems in situations of certainty, risk and uncertainty, etc. The features and advantages of the language are most evident when we simulate and solve mathematical programming problems of medium and large dimension, in particular, the compactness of the representation of the mathematical model and the ease of scaling the modeled tasks. A structured representation of the problem model allows us to separate the optimization model from the data, to enter the sections of the preliminary calculations (e.g., average and dispersion) and of set the initial conditions. The modeling language viewed in the article enables in one description to specify the variance of the solution of tasks, to allocate sub-models and from them to form the necessary combinations, each of which is solved as a single task. The language includes means of programming which allow to enter data directly into the model or to obtain them from external sources (text files, spreadsheets, databases, and custom applications), and to transfer the results of solution on the screen or in external files, databases, etc. The examples in the article demonstrate the flexibility of the LINGO modeling language, and the compactness of received models.Keywords: modeling language, tasks of mathematical programming, LINGO.

Keywords: modeling language, tasks of mathematical programming, LINGO.

Математическое программирование как совокупность методов поиска оптимальных решений находит практическое применение в различных сферах человеческой деятельности: в науке и бизнесе, в экономике и финансах, в строительстве и сельском хозяйстве, в военном деле и сфере услуг, и т.д. [1–3]. С его помощью решаются задачи прогнозирования, исследований, проектирования и конструирования, планирования, размещения, организации и производства продукции, логистики и безопасности, и др. [4–9, 11, 15].

Математические модели таких задач оптимизации содержат от десятков до многих тысяч переменных и условий (ограничений), т.е. относятся к задачам средней и большой размерности [10]. Поэтому при использовании компьютерных программ возникает проблема представления в них математической модели решаемой задачи. Например, в простейшей транспортной задаче, включающей 50 пунктов отправления и 1000 пунктов назначения, модель будет содержать 50 тыс. переменных и 1050 функциональных ограничений. В известной задаче коммивояжера с n городами, которая сегодня находит практическое применение, число переменных равно $n^2 + n - 1$, число равенств – $2n$ и число неравенств, исключающее подциклы, – $(n - 1)^2 - (n - 1)$ [13]. Еще бóльшая размерность моделей присуща многоиндексным транспортным и производственным системам [10, 11] .

На наш взгляд, указанная проблема (наряду с другими подобными) довольно успешно и элегантно решена в пакете математического программирования LINGO. Этот пакет имеет встроенный мощный и гибкий язык математического описания задач оптимизации, позволяющий представлять модель задачи в компактной форме с последующим решением встроенным решателем или решателями. При этом собственно математическая модель не зависит от размера задачи: так, модель вышеприведенной транспортной задачи на языке LINGO будет содержать выражения целевой функции в виде короткой строки и двух ограничений, по одному для пунктов отправления и пунктов назначения. При изменении числа пунктов в любую сторону эти выражения не изменятся. Поэтому масштабирование задачи в LINGO не вызывает затруднений (как правило, оно касается только данных).

Далее рассмотрим основные конструкции языка моделирования LINGO. Сразу заметим, что для решения небольших задач можно обойтись без использования специального языка, представив модель в обычном математическом виде со всеми числовыми коэффициентами, как, например, в пакете LINDO (предшественник LINGO) [12].

Первый принцип, обеспечивающий компактность модели, – работа с наборами или множествами (sets) объектов, которые могут иметь атрибуты. Второй принцип – отделение данных от математического описания. Эти принципы реализуются через структурированность модели задачи. Модель большой системы (задачи) обычно имеет три секции или раздела (sections): SETS, DATA и математическая модель (ММ). В первом разделе описываются структуры данных через задание множеств и, возможно, их атрибутов. Каждое множество – это совокупность подобных объектов. В разделе DATA приводятся данные задачи или указываются их источник (текстовый файл, электронная таблица, база данных или приложение) и направление вывода результатов решения. Раздел ММ содержит целевую функцию и ограничения задачи. Каждый из этих разделов может быть представлен и более одного раза в разных частях модели, но при этом обязательно предшествование описания объектов и данных ссылкам на них. При необходимости модель задачи дополняется вспомогательными разделами INIT и CALC, в которых приводятся соответственно начальные значения (например, стартовые точки) и выражения, обеспечивающие обработку исходных данных (например, вычисление средних, дисперсий и т.п.). Все разделы, за исключением ММ, выделяются в модели задачи ключевыми словами начала и конца раздела, содержащими имя раздела.

LINGO воспринимает два вида множеств: первичные или примитивные (primitive) и вторичные или производные (derived). Множества derived образуются из одного или нескольких других множеств, выступающих в роли родителей.

Примеры задания множеств primitive:

```
SETS:
 SUPPLIERS /1..N/: A;
 CONSUMERS /1..9/: B;
 PRODUCT /Pr1, Pr2, Pr3/: price, Q;
 QUALITY: level;
 FIRMS;
ENDSETS
```

Здесь определены 5 множеств, которые задают N поставщиков с атрибутом A – количество товара, 9 потребителей с атрибутом B – величина потребности, 3 продукта с ценой и количеством; элементы последних двух множеств должны быть перечислены в секции DATA. Явное задание числа поставщиков N отнесено к предшествующей секции данных.

Производное множество может быть образовано из одного примитивного, как его подмножество, с помощью задания условий (фильтра) на его атрибуты, которые записываются после вертикальной черты, например:

```
TOP_PRODUCT (PRODUCT) | price (&1) #GE# 2000;
```

В этом определении символ &1 как индекс родительского множества указывает на принадлежность атрибута первому, здесь единственному множеству PRODUCT, #GE# – логический оператор (>=). В условиях могут использоваться также операторы #EQ# (=), #NE# (≠), #GT# (>), LT (<), #LE# (<=) и общепринятые логические операции (#AND# и т.д.).

В общем случае производное множество вводится следующим образом:

```
set_name (parent_set_list) [filter] [: attribute_list];
```

где обязательным является только список родительских множеств.

Пример:

```
SETS:
 SUPPLIERS;
 CONSUMERS;
 NET (SUPPLIERS, CONSUMERS): cost, X;
ENDSETS
```

Здесь производное множество NET имеет общие атрибуты: стоимость перевозки и количество груза, перевозимое от поставщиков к потребителям.

Различают плотные производные множества (dense) и разреженные (sparse). Плотное множество содержит все сочетания элементов родительских множеств, т.е. определяется их декартовым произведением. Для образования разреженного множества применяется фильтр (условия) или явное перечисление всех входящих в него элементов.

Раздел DATA содержит значения атрибутов множеств и элементы тех множеств, которые не раскрыты в разделе SETS. При явном представлении данных используются выражения:

```
attribute_list = value_list;
set_name = member_list;
```

Оба варианта показаны в следующем примере:

```
SETS:
SET1 / M1, M2, M3/: X, Y;
SET2: Z;
ENDSETS
DATA:
X = 8 12 3;
Y = 74 45 66;
SET2=A, B, C, D;
     Z=91 47 23 82;
ENDDATA
```

Если все данные или их часть берутся из внешних источников, то в секции DATA они описываются с помощью функций @TEXT, @OLE, @ODBC или @POINTER для обмена соответственно с файлами, электронными таблицами, БД или приложениями.

Эффективность языка моделирования в LINGO определяется возможностью применять операцию и/или соотношения последовательно ко всем элементам множества с помощью одного операторного выражения. Для реализации этой возможности LINGO имеет специальные функции, называемые *set looping functions* (*SLF*). К ним относятся функции @FOR, @SUM, @MIN, @MAX и @PROD. Первая используется для генерации ограничений для всех членов множества, последняя – для вычисления произведения, остальные функции пояснений не требуют. Допускается любая вложенность *SLF*-функций за исключением @FOR, которая может быть вложенной только в другую функцию @FOR.

Синтаксис *SLF*-функций:

```
@loop_function ( setname [ ( set_index_list)[ | conditional_qualifier]] : expression_list);
```

Как видно, обязательными аргументами функции являются множество и выражения, применяемые ко всем элементам этого множества.

При наличии условий действие функции будет распространяться только на те члены множества, которые им удовлетворяют.

Для описания прямых ограничений на переменные применяются следующие функции:

@GIN – задает целочисленный тип переменных;

@BIN – задает бинарный тип переменных;

@FREE – придает вещественной переменной неограниченность по знаку;

@BND – накладывает двухсторонние ограничения;

@SOS1 – из всех бинарных переменных, описанных этой функцией, самое большее – это одна может быть равна 1;

@SOS2 – не более двух бинарных переменных могут равняться 1, и если ровно 2 переменные ненулевые, то они будут смежными;

@SOS3 – точно только одна бинарная переменная равна 1;

@CARD – из всех переменных, описанных этой функцией, не более *N* переменных могут быть больше нуля;

@SEMIC – переменная может либо равняться нулю, либо быть больше некоторой константы.

Функция @SOS2 применяется, например, при описании кусочно-линейных зависимостей, а функция @SEMIC – для зависимостей с разрывом в нуле, например, при учете в модели фиксированных затрат на производство или перевозки.

Функции @SOS и @CARD имеют общий синтаксис:

@function('set_name', *variable*),

где set_name – уникальное имя списка переменных, к которым относится действие этой функции (таких записей столько, сколько переменных в списке). При использовании функции @CARD записи по переменным дополняются особой записью (одной на список), в которой вместо переменной указывается целое число *N*.

При моделировании некоторых ситуаций возникает необходимость преобразовывать текущий индекс так, чтобы его значение принадлежало заданному диапазону. Такое преобразование выполняет функция @WRAP, имеющая 2 аргумента: *INDEX* и *LIMIT* (фактический индекс и верхняя граница диапазона). Функция возвращает значение из интервала [1, *LIMIT*]. Так, при планировании на 3 года текущий индекс может представлять порядковый номер месяца в плановом периоде. Если он равен, например, 27, функция @WRAP(27, 12) вернет значение 3, т.е. месяц март.

Еще одна сильная сторона языка моделирования LINGO – это возможность реализации в одной модели вариантных решений (например, по разным наборам целевых функций и ограничений), декомпозиции большой задачи и т.п. Для этого в разделе ММ выделяются модели подзадач (подмодели) следующим образом:

```
submodel name_sabmodel:
подмодель;
endsubmodel.
```

Подмодель может включать только целевую функцию, только ограничения или то и другое. Комбинации подмоделей указываются в функции @SOLVE, которая записывается в разделе CALC. Она объединяет подмодели в единую модель и запускает ее решение.

Рассмотрим пример построения модели типичной производственной задачи на языке моделирования LINGO. Пусть некоторая фирма имеет n типов универсального оборудования, на котором может производить определенный ассортимент изделий (m видов). Типы оборудования отличаются временем t_{ij}, необходимым на изготовление одного i-го изделия на j-м оборудовании. При получении заказа на изделия его необходимо распределить по оборудованию так, чтобы выполнить за минимальное время. Следует учесть, что заказ может включать не весь ассортимент изделий, и по технологическим причинам число видов изделий, производимых на одном типе оборудовании за время выполнения заказа, ограничено величиной k_j.

Исходная математическая модель этой задачи имеет вид:

$$Time = \max_j T_j \Rightarrow \min,$$

$$T_j = \sum_{i=1}^{m} t_{ij} x_{ij},\ j = \overline{1,n},$$

$$\sum_{j=1}^{n} x_{ij} = b_i,\ i = \overline{1,m},$$

$$\sum_{i=1}^{m} y_{ij} \le k_j,\ j = \overline{1,n},$$

$$\forall ij\ \ 0 \le x_{ij} \le M y_{ij}, \text{ целые}; \quad y_{ij} - \text{бинарные},$$

где x_{ij} – количество изделий i-го типа, производимых на j-м оборудовании, b_i – заказанное количество изделий i-го типа; y_{ij} = 1, если на j-м оборудовании производятся i-е изделия, иначе $y_{ij} = 0$; M – большое положительное число. Эта модель содержит $2nm + n$ переменных, $n + m$ равенств и $n + nm$ неравенств, что свидетельствует о большой размерности задачи.

На языке моделирования, рассмотренном выше, модель этой же задачи записывается в виде:

```
MODEL:
 sets:
  PRODUCT: order;
  EQUIPMENT /1..3/: TWORK, K;
  PR_EQ (PRODUCT, EQUIPMENT): T, X;
 endsets
 data:
   PRODUCT=pr1 .. pr4;
   order= 75 123 0 62;
  T = 7 11 9
      14 8 10
      12 7 13
 10 9 11;
  K = 2 1 3;
 enddata
  MIN = TIME;
  @FOR(PRODUCT(I)|order #NE# 0: @SUM(EQUIPMENT(J):
X(I,J))=order);
  @FOR(EQUIPMENT(J): @SUM(PRODUCT(I)|order #NE# 0:
T(I,J)*X(I,J))- TWORK(J)=0);
  TIME = @MAX(EQUIPMENT: TWORK);
  @FOR(PR_EQ(I,J)|order(I) #NE# 0: @GIN(X));
  @FOR( EQUIPMENT(J): @CARD( 'name'+ EQUIPMENT(J),
K);
     @FOR( PRODUCT(I)|order #NE# 0:  @CARD('name'+
EQUIPMENT(J),         X(I,J))));
END
```

Здесь введены два первичных множества и одно производное, атрибутом PRODUCT является величина заказа, а атрибутами множества EQUIPMENT – время работы оборудования и число видов изделий. Заметим, что ограничения по числу видов изделий на одном типе оборудования k_j описаны не неравенствами, а функцией @CARD. Условие order #NE# 0 исключает из модели ограничения и переменные, соответствующие отсутствующим в заказе видам изделий. Очевидно, что выражения целевой функции и ограничений в этой модели не зависят от значений n и m, и при их изменении потребуется изменить только данные. Для примера мы взяли $n = 3$ и $m = 4$, в этом случае LINGO сгенерирует для решателя развернутую математическую модель, соответствующую введенным данным, в следующем виде:

```
    MODEL:
     [_1] MIN= TIME;
     [_2] X_PR1_1 + X_PR1_2 + X_PR1_3 = 75;
     [_3] X_PR2_1 + X_PR2_2 + X_PR2_3 = 123;
     [_4] X_PR4_1 + X_PR4_2 + X_PR4_3 = 62;
     [_5] 7 * X_PR1_1 + 14 * X_PR2_1 + 10 * X_PR4_1 -
TWORK_1 = 0;
     [_6] 11 * X_PR1_2 + 8 * X_PR2_2 + 9 * X_PR4_2 -
TWORK_2 = 0;
     [_7] 9 * X_PR1_3 + 10 * X_PR2_3 + 11 * X_PR4_3 -
TWORK_3 = 0;
     [_8] TIME = @SMAX( TWORK_1, TWORK_2, TWORK_3);
     @CARD( 'NAME1', X_PR1_1); @CARD( 'NAME1',
X_PR2_1);
     @CARD( 'NAME1', X_PR4_1); @CARD( 'NAME1', 2);
@CARD( 'NAME2', X_PR1_2);
     @CARD( 'NAME2', X_PR2_2); @CARD( 'NAME2',
X_PR4_2); @CARD( 'NAME2', 1);
     @CARD( 'NAME3', X_PR1_3); @CARD( 'NAME3',
X_PR2_3);
     @CARD( 'NAME3', X_PR4_3); @CARD( 'NAME3', 3);
     @GIN( X_PR1_1); @GIN( X_PR1_2); @GIN( X_PR1_3);
@GIN( X_PR2_1);
     @GIN( X_PR2_2); @GIN( X_PR2_3); @GIN( X_PR4_1);
@GIN( X_PR4_2);
     @GIN( X_PR4_3);
     END
```

Как и следовало ожидать, в этой модели присутствуют только те ограничения и переменные, которые соответствуют реальному заказу. Нетрудно видеть, что построенная модель задачи является нелинейной. Простым преобразованием ее можно привести к линейному виду: достаточно исключить из целевой функции взятие максимума, заменив его неравенствами TIME $\geq T_j$, $j = 1..n$.

Интересно сравнить решения задачи, полученные по линейной и нелинейной модели. С этой целью, не затрагивая разделы SETS и DATA, изменим раздел MM, заменив его подмоделями, и добавим раздел CALC, где сформируем оба варианта модели и решим их. В результате разделы MM и CALC принимают вид:

```
. . .
enddata
 submodel OBJ_1:
  MIN = TIME1;
  TIME1 = @MAX(EQUIPMENT:  TWORK);
 endsubmodel
 submodel OBJ_2:
  MIN = TIME2;
  @FOR(EQUIPMENT: TIME2>=TWORK);
 endsubmodel
 submodel CONSTR:
  @FOR(PRODUCT(I)|order  #NE#  0:  @SUM(EQUIPMENT(J):
X(I,J))=order);
  @FOR(EQUIPMENT(J):  @SUM(PRODUCT(I)|order  #NE#  0:
T(I,J)*X(I,J))- TWORK(J)=0);
  @FOR(PR_EQ(I,J)|order(I) #NE# 0: @GIN(X));
  @FOR( EQUIPMENT(J): @CARD( 'name'+ EQUIPMENT(J), K);
     @FOR(  PRODUCT(I)|order #NE# 0:   @CARD('name'+
EQUIPMENT(J), X(I,J))));
 endsubmodel
 calc:
  @SOLVE(OBJ_2, CONSTR);
  @SOLVE(OBJ_1, CONSTR);
 Endcalc
```

Как видно из раздела CALC, первой решается линейная модель, а затем нелинейная. Оба решения выводятся в одном отчете:

Global optimal solution found.

Objective value:	745.0000
Objective bound:	745.0000
Infeasibilities:	0.000000
Extended solver steps:	7
Total solver iterations:	141

Model Class: **MILP**

Variable	Value	Reduced Cost
TIME1	0.000000	0.000000
TIME2	745.0000	0.000000
TWORK(1)	745.0000	0.000000
TWORK(2)	744.0000	0.000000
TWORK(3)	740.0000	0.000000
X(PR1, 1)	75.00000	7.000000
X(PR1, 2)	0.000000	0.000000
X(PR1, 3)	0.000000	0.000000
X(PR2, 1)	0.000000	14.00000
X(PR2, 2)	93.00000	0.000000
X(PR2, 3)	30.00000	0.000000
X(PR3, 1)	0.000000	0.000000
X(PR3, 2)	0.000000	0.000000
X(PR3, 3)	0.000000	0.000000
X(PR4, 1)	22.00000	10.00000
X(PR4, 2)	0.000000	0.000000
X(PR4, 3)	40.00000	0.000000

Local optimal solution found.

Objective value:	767.0000
Objective bound:	767.0000
Infeasibilities:	0.000000
Extended solver steps:	29
Total solver iterations:	2885

Model Class: **PINLP**

Variable	Value
TIME1	767.0000
TIME2	745.0000
TWORK(1)	767.0000
TWORK(2)	760.0000
TWORK(3)	766.0000
X(PR1, 1)	21.00000
X(PR1, 2)	0.000000
X(PR2, 1)	0.000000
X(PR2, 2)	95.00000
X(PR2, 3)	28.00000
X(PR3, 1)	0.000000
X(PR3, 2)	0.000000
X(PR3, 3)	0.000000
X(PR4, 1)	62.00000
X(PR4, 2)	0.000000
X(PR4, 3)	0.000000

Отчет показывает, что LINGO воспринял первую модель как модель смешанного (непрерывно-целочисленного) линейного программирования (MILP), а вторую – как модель полностью целочисленного нелинейного программирования (PINLP). Соответственно в первом случае найден глобальный минимум, а во втором только локальный. При этом для решения нелинейной модели потребовалось 2885 итераций вместо 141 для линейной.

Приведенный пример продемонстрировал, с одной стороны, возможности языка моделирования LINGO, с другой – важность преобразования модели к более простому классу.

Рамки статьи не позволяют раскрыть весь спектр возможностей языка моделирования LINGO: помимо рассмотренных конструкций язык включает множество математических и финансовых функций, функции программирования хода решения и оформления отчетов, линейку функций для моделирования непрерывных и дискретных случайных величин и их вероятностных распределений, функции моделирования задач стохастического программирования и др. [14]. Несмотря на это, приведенный в статье материал наглядно показывает особенности и существенные преимущества этого языка при моделировании и решении задач оптимизации как средней, так и большой размерности.

Библиографический список

1. Мину М. Математическое программирование. Теория и алгоритмы: пер. с фр. – М.: Наука, 1990. – 486 с.

2. Карманов В.Г. Математическое программирование: учеб. пособие для вузов. – 6-е изд., испр. – М.: Физматлит, 2008. – 263 с.

3. Базара М., Шетти К. Нелинейное программирование. Теория и алгоритмы: пер. с англ. / под ред. Д.Б. Юдина. – М.: Мир, 1982. – 583 с.

4. Грешилов А.А. Прикладные задачи математического программирования: учеб. пособие. – 2-е изд., доп. – М.: Логос, 2006. – 286 с.

5. Пыткин А. Н. Моделирование механизма территориального планирования промышленного сектора экономики региона. – Екатеринбург: Изд-во Ин-та экономики УрО РАН, 2009. – 167 с.

6. Уайлд Д. Оптимальное проектирование: пер. с англ. – М.: Мир, 1981. – 272 с.

7. Гилл Ф., Мюррей У., Райт М. Практическая оптимизация: пер. с англ. – М.: Мир, 1985. – 509 с.

8. Месарович М., Мако Д., Такахара И. Теория иерархических многоуровневых систем: пер. с англ. – М.: Мир, 1973. – 344 с.

9. Реклейтис Г., Рейвиндран А., Рэгсдел К. Оптимизация в технике: в 2 кн.: пер. с англ. – М.: Мир, 1986. – Кн. 2. – 320 с.

10. Лэсдон Л.С. Оптимизация больших систем. – М.: Наука, 1975. – 432 с.

11. Раскин Л.Г., Кириченко И.О. Многоиндексные задачи линейного программирования (теория, методы и приложения). – М.: Радио и связь, 1982. – 282 с.

12. Гольдштейн А.Л. Оптимизация в LINDO. – Пермь: Изд-во Перм. гос. техн. ун-та, 2000. – 87 с.

13. Вагнер Г. Основы исследования операций: пер. с англ.: в 3 т. – М.: Мир, 1972. – Т. 2. – 1973. – 488 с.

14. Сайт LINDO SYSTEMS Inc [Электронный ресурс]. – URL: http://www.lindo.com (дата обращения: 14.03.2016).

15. Михалевич В.С., Трубин В.А., Шор Н.З. Оптимизационные задачи производственно-транспортного планирования: Модели, методы, алгоритмы. – М.: Наука, 1986. – 264 с.

References

1. Minu M. Matematicheskoe programmirovanie. Teoriia i algoritmy [Mathematical programming. Theory and algorithms]. Moscow: Nauka, 1990. 486 p.

2. Karmanov V.G. Matematicheskoe programmirovanie [Mathematical programming]. Moscow: Fizmatlit, 2008. 263 p.

3. Bazara M., Shetti K. Nelineinoe programmirovanie.Teoriia i algoritmy [Nonlinear Programming. Theoryand algorithms]. Moscow: Mir, 1982. 583 p.

4. Greshilov A.A. Prikladnye zadachi matematicheskogo programmirovaniia [Mathematical programming application tasks]. Moscow: Logos, 2006. 286 p.

5. Pytkin A.N. Modelirovanie mekhanizma territorial'nogo planirovaniia promyshlennogo sektora ekonomiki regiona [The urban planning mechanism of the region economy industrial sector modeling]. Ekaterinburg: Institut ekonomiki Ural'skogo otdeleniia Rossiiskoi akademii nauk, 2009. 167 p.

6. Uaild D. Optimal'noe proektirovanie [Optimum design]. Moscow: Mir, 1981. 272 p.

7. Gill F., Miurrei U., Rait M. Prakticheskaia optimizatsiia [Practical optimization]. Moscow: Mir, 1985. 509 p.

8. Mesarovich M., Mako D., Takakhara I. Teoriia ierarkhicheskikh mnogourovnevykh sistem [Multilevel hierarchy system theory]. Moscow: Mir, 1973. 344 p.

9. Rekleitis G., Reivindran A., Regsdel K. Optimizatsiia v tekhnike [Equipment optimization]. Moscow: Mir, vol. 2, 1986, 320 p.

10. Lesdon L.S. Optimizatsiia bol'shikh sistem [Large systems optimization]. Moscow: Nauka, 1975. 432 p.

11. Raskin L.G., Kirichenko I.O. Mnogoindeksnye zadachi lineinogo programmirovaniia (teoriia, metody i prilozheniia) [Multi-indexes problems of the linear programming (theory, methods and applications)]. Moscow: Radio i sviaz', 1982. 282 p.

12. Gol'dshtein A.L. Optimizatsiia v LINDO [Optimization in LINDO]. Permskii gosudarstvennyi tekhnicheskii universitet, 2000. 87 p.

13. Vagner G. Osnovy issledovaniia operatsii [Operation performance analysis bases]. Moscow: Mir, 1972, vol. 2, 1973. 488 p.

14. Sait LINDO SYSTEMS Inc [Web site LINDO SYSTEMS Inc], available at: http://www.lindo.com (accessed 14 March 2016).

15. Mikhalevich V.S., Trubin V.A., Shor N.Z. Optimizatsionnye zadachi proizvodstvenno-transportnogo planirovaniia: Modeli, metody, algoritmy [Optimization tasks of the industrial-transportation planning and processing. Models, methods and algorithms]. Moscow: Nauka, 1986. 264 p.

Сведения об авторе

Гольдштейн Аркадий Леонидович (Пермь, Россия) – кандидат технических наук, профессор кафедры информационных технологий и автоматизированных систем Пермского национального исследовательского политехнического университета (614990, Пермь, Комсомольский пр., 29, e-mail: gal@pstu.ru).

About the author

Goldshtein Arkady Leonidovich (Perm, Russian Federation) is a Ph.D. in Technical Sciences, Professor at the department of information technology and automated systems of Perm National Research Polytechnic University (614990, Perm, 29, Komsomolsky pr., e-mail: gal@pstu.ru).

Получено 20.04.2016