УДК 658.52.011.56

КОНСТРУИРОВАНИЕ И ИССЛЕДОВАНИЕ АЛГОРИТМОВ РЕШЕНИЯ ЗАДАЧИ О РЮКЗАКЕ

С.А. Канцедал, проф., д.т.н., Западнодонбасский институт Межрегиональной академии управления персоналом, г. Павлоград, М.В. Костикова, доц., к.т.н., И.В. Скрипина, ст. преп., Харьковский национальный автомобильно-дорожный университет

Аннотация. *Изложены и обоснованы основные понятия и возможности решения задачи о рюкзаке точными методами ветвей и границ и методами динамического программирования. Описываются методы и алгоритмы решения задачи. Приводятся сравнительные характеристики алгоритмов, которые получены экспериментальным путём. Даются рекомендации по их практическому применению.*

Ключевые слова: *задача о рюкзаке, алгоритм, динамическое программирование, ветви и границы, стратегия.*

КОНСТРУЮВАННЯ Й ДОСЛІДЖЕННЯ АЛГОРИТМІВ РІШЕННЯ ЗАДАЧІ ПРО РЮКЗАК

С.А. Канцедал, проф., д.т.н., Західнодонбаський інститут Міжрегіональної академії управління персоналом, м. Павлоград, М.В. Костікова, доц., к.т.н., І.В. Скрипіна, ст. викл., Харківський національний автомобільно-дорожній університет

Анотація. *Викладено та обґрунтовано основні поняття й можливості рішення задачі про рюкзак точними методами гілок і границь та методами динамічного програмування. Описуються методи та алгоритм розв'язання задачі. Наводяться порівняльні характеристики алгоритмів, одержані експериментальним шляхом. Даються рекомендації до їх практичного застосування.*

Ключові слова: *задача про рюкзак, алгоритм, динамічне програмування, гілки і границі, стратегія.*

DESIGN AND EXAMINATION OF ALGORITHMS FOR SOLVING THE KNAPSACK PROBLEM

S. Kantsedal, Prof., D. S. (Eng.), The West Donbass Institute of Interregional Academy of Personnel Management, the town of Pavlograd, M. Kostikova, Assoc. Prof., Ph. D. (Eng.), I. Skrypina, senior teacher, Kharkov National Automobile and Highway University

Abstract. *The use of methods of branches and boundaries as well as the methods of dynamic programming at solving the problem of «knapsack» is grounded. The main concepts are expounded. The methods and algorithms development for solving the above specified problem are described. Recommendations on practical application of constructed algorithms based on their experimental investigation and carrying out charactheristics comparison are presented.*

Key words: *knapsack problem, algorithm, dynamic programming, branch and bound, strategy.*

Введение

Впервые сформулированная американским математиком Д.Б. Данцигом задача о рюкзаке (ранце) (англ. *Knapsack problem*) – одна из *NP*-полных задач комбинаторной оптимизации, популярность которой вызвана большим количеством её приложений, поскольку многие из реально возникающих задач описываются в рамках данной модели. Основные сферы применения находятся в областях планирования и управления производственными и транспортными системами. Название своё получила от максимизационной задачи укладки как можно большего числа ценных вещей в рюкзак при условии, что общий объём (или вес) всех предметов, способных поместиться в рюкзак, ограничен. Непосредственно термин «рюкзак» может быть интерпретирован достаточно широко. Данная задача и её варианты широко используются для моделирования большого числа практических задач.

Анализ публикаций

Задачи о рюкзаке и её модификации часто возникают в экономике, прикладной математике, криптографии, генетике и логистике для нахождения оптимальной загрузки транспорта (автомобиля, контейнера, самолёта, поезда, трюма корабля) или склада. Поэтому разработке методов решения задачи, и в первую очередь эффективных, уделено достаточно много внимания [1–7].

В литературе можно встретить обоснование и анализ различных методов решения общей задачи линейного программирования, задачи целочисленного программирования, потоковых задач, ряда задач на графах, задач о матроидах [1]. Приводятся также задачи дискретного программирования, заключающиеся в нахождении условных экстремумов на конечных множествах [2], задачи геометрической оптимизации (раскрашивание графа, реализация графа с минимальным числом пересечений, наиболее плотная упаковка) [4].

Исследуются как точные, так и приближённые алгоритмы решения задачи.

В настоящее время обсуждается возможность реализации задачи на основе физической системы квантового компьютера для решения задачи об упаковке рюкзака [5].

Очень часто при решении задач такого класса на практике рассматривается двухкритериальная задача о рюкзаке [6].

Цель и постановка задачи

Существуют различные точные и приближённые алгоритмы решения задачи о рюкзаке. К точным алгоритмам относятся: полный перебор; метод ветвей и границ динамического программирования.

В общем виде задача о рюкзаке формулируется следующим образом. Перед походом в рюкзак, вместимостью не более A единиц веса, из набора $I = \{1, 2, \ldots, i, \ldots, n\}$ предметов, каждый весом a_i и «ценностью» c_i, необходимо положить те предметы x_i, $i \in I$, которые максимизируют суммарную «ценность» груза и помещаются по весу в рюкзак.

В математической форме задача представляется следующим образом: требуется найти x_i, $i \in I$, доставляющее

$$\max \sum_{i \in I} c_i x_i = W, \qquad (1)$$

при условиях

$$\sum_{i \in I} a_i x_i \le V, \qquad (2)$$

$$x_i = \begin{cases} 1, & \text{если предмет помещён в рюкзак,} \\ 0, & \text{если предмет не помещён в рюкзак.} \end{cases} \qquad (3)$$

Таким образом, задача о рюкзаке – это задача целочисленного линейного программирования с булевыми переменными. Её решение достигается на некотором подмножестве 2^n комбинаций, формируемом различными наборами переменных x_i, удовлетворяющих ограничению (2). Известно, что задача является *NP*-полной, т. е. для неё теоретически не существует полиномиального по времени решения алгоритма [1]. Среди экспоненциальных алгоритмов наиболее подходящими являются те, которые реализуют схему ветвей и границ, метод динамического программирования, аддитивный алгоритм Балаша [2], а также другие разработанные алгоритмы. Хотя эти алгоритмы и оценены по вычислительной сложности, вместе с тем они не имеют практических оценок решения

задач, оценок работы алгоритмов в среднем, что очень важно для их применения.

Цель работы состоит в конструировании и экспериментальном исследовании трёх алгоритмов, предназначенных для решения задачи о рюкзаке. Два из них – метод ветвей и границ и динамического программирования – позволяют получить точные решения, третий метод – эвристический – даёт приближенные решения задачи. Приводятся сравнительные характеристики алгоритмов, на основании чего даются рекомендации к их практическому применению.

Разработка алгоритмов

Как известно, в методе ветвей и границ имеется два момента алгоритмизации, определяемых спецификой задачи: разбиение исходного множества комбинаций на подмножества с дальнейшим выбором подмножества для очередного разбиения и вычисления нижних (верхних) границ (оценок) значений оптимизируемой функции на подмножествах. Разбиение множества на подмножества называется ветвлением, а выбор подмножества для разбиения – его стратегией. Вычисление ошибок толкуют как решение оценочных задач. Наглядным результатом ветвления и решения оценочных задач является n-ярусное корневое дерево поиска решений с оценками вершин подмножеств каждого яруса. Оценка вершины последнего яруса – рекорд – представляет собой текущее значение оптимизируемой функции, которое далее сравнивается с оценками вершин предшествующих ярусов, в результате чего неперспективные для ветвления подмножества отсеиваются, а перспективные разбиваются, дополняя дерево решений. Алгоритм заканчивает работу тогда, когда будут сравнены с рекордом все текущие и вновь порождаемые оценки вершин. Практика показывает, что более эффективными являются те алгоритмы, которые строят бинарные деревья решений, т. е. реализуют разбиение каждого очередного множества на два подмножества, выбор подмножества для ветвления осуществляют по максимальной (минимальной) оценке оптимизируемой функции, оценочные задачи формируются так, что более точно вычисляются оценки. Поэтому при разработке алгоритма и наличии выбора следует придерживаться указанных правил.

Что касается рассматриваемой задачи, то процесс разбиения очередного множества осуществляется на два подмножества, первое из которых содержит комбинации компонент вектора с $x_i = 0$, второе – с $x_i = 1$. Стратегия ветвления состоит в том, чтобы выбирать очередное подмножество для разбиения по максимальной оценке верхней границы оптимизируемой функции. В результате получаем бинарное дерево поиска решений, изображённое на рис. 1.

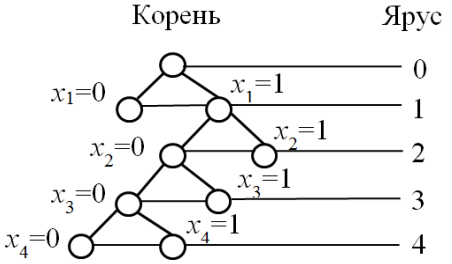

Рис. 1. Бинарное дерево поиска решений

Для вычисления оценок верхних границ оптимизируемой функции на подмножествах формулируется следующая оценочная задача: найти

$$Q = \max \sum_{i \in I} c_i \cdot x_i, \qquad (4)$$

при условиях

$$\sum_{i \in I} a_i \cdot x_i \le V, \qquad (5)$$

$$0 \le x_i \le 1, \; i \in I. \qquad (6)$$

Иными словами, для некоторого x_i допускается, что оно не булево, а лежит в интервале $[0,1]$. Такая релаксация строгости условий задачи приводит к тому, что оценка Q для каждой вершины дерева поиска оказывается больше W, т. е. она действительно является верхней границей оптимизируемой функции на подмножествах комбинаций.

В то же время оценочная задача легко решаема. Согласно [3] следует найти «ценности» единиц весов грузов $\frac{c_i}{a_i}$, $i = 1, 2, ..., n$ и упорядочить их по убыванию $\frac{c_1}{a_1} \ge \frac{c_2}{a_2} \ge \ldots \ge \frac{c_n}{a_n}$. Тогда все $x_i \in I$, выбираемые в порядке по-

следовательности «ценностей» и такие, что $\sum_{i=1}^{k} a_i < V$, полагаются равными единице. Очередное значение x_{k+1} такое, что $\sum_{i=1}^{k+1} a_i \cdot x_i > V$ вычисляется по выражению $x_{k+1} = (V - \sum_{i=1}^{k} a_i) / a_{k+1}$, т. е. ищется значение той переменной x_i, $i \in I$, которая полагается не булевой. Остальные x_i, $i = k+2, \ldots, n$ принимается равными нулю. В результате получаем оценку $Q = \sum_{i=1}^{k+1} c_i \cdot x_i$ и ограничение $\sum_{i=1}^{k+1} a_i < V$. Пусть для ветвления выбрана вершина r-го яруса дерева поиска с оценкой Q_r и значением V_r, которые определены последовательностью x_i, $i = 1, 2, \ldots, r$. Тогда оценочные задачи для вершины $r+1$-го яруса должны быть сформулированы так:
найти

$$Q_{r+1} = \max\left(Q_r + \sum_{i=r+1}^{n} c_i \cdot x_i \right),$$

при условиях

$$\sum_{i=r+1}^{n} a_i \cdot x_i \le V - V_r ,$$

$$0 \le x_i \le 1,\ i \in I .$$

Алгоритм, реализующий описанный метод, приведён ниже.

Шаг 1. Установить n; положить $k = 0$, рекорд $R = 0$.
Шаг 2. Положить $k = k + 1$.
Шаг 3. Положить $x_k = 0$, вычислить оценку Q_k при $x_k = 0$ и запомнить Q_k.
Шаг 4. Положить $x_k = 1$, вычислить оценку Q_k при $x_k = 1$.
Шаг 5. Выбрать вершину для ветвления v_x с большей оценкой Q_k.
Шаг 6. Если $Q_k \le R$, перейти к шагу 9, иначе перейти к шагу 7.
Шаг 7. Если $k < n$, запомнить большее Q_k, соответствующее x_k, и вернуться к шагу 2; иначе перейти к шагу 8.
Шаг 8. Положить R равным большему Q_k. Запомнить вектор X.
Шаг 9. Положить $k = k - 1$.
Шаг 10. Если $k = 0$, остановиться; иначе перейти к шагу 11.
Шаг 11. Выбрать запомненное Q_k.
Шаг 12. Если $Q_k \le R$, вернуться к шагу 9, иначе вернуться к шагу 3.

Метод динамического программирования вначале разрабатывался Р. Беллманом и его учениками для решения задач управления динамическими объектами, движение которых (переход из состояния в состояние) описывалось дифференциальными уравнениями, осуществлялось под воздействием управлений, а оптимизируемыми функциями являлись интегральные функционалы. Позже он был распространён для решения некоторых классов задач статической оптимизации, в частности, задач комбинаторного характера.

Метод предполагает, что процесс управления может быть представлен как последовательный N-шаговый (N-стадийный) процесс принятия решений. При этом выработка решений на очередных этапах не оказывает влияние на величину оптимизируемой функции, достигнутой на предшествующих шагах, т. е. рассматриваются системы без последствия (обратной связи).

Если указанные условия выполняются, то оптимальное управление может быть построено, если следовать принципу оптимальности: оптимальная стратегия обладает тем свойством, что каковы бы ни были непосредственно предшествующее состояние и управление, последующее управление должно быть оптимальным по отношению к этому состоянию. Иными словами, в каждом состоянии по отношению к нему необходимо действовать оптимально.

Чаще всего при построении оптимальной стратегии управления вычисления начинают в порядке, обратном течению N-шагового процесса принятия решений, т. е. с конца. При этом обязательным при решении всех задач является построение на каждом шаге функции Беллмана: экстремального значения функционала по управлению в зависимости от состояния системы. В том случае, когда в

наличии много возможных состояний либо много допустимых управлений, либо имеет место то и другое, вычисление функции Беллмана может оказаться слишком трудоёмким и в целом метод окажется неэффективным. Наиболее просто функция Беллмана вычисляется для первого шага. Для всех остальных шагов она определяется на основании функционального уравнения, связывающего очередной и предшествующий шаги процесса принятия решений. Вывод функционального уравнения, фактически рекуррентного соотношения, специфичен для каждой задачи и во многих случаях может представлять определённые трудности.

В рассматриваемой задаче оптимальную стратегию представляет набор значений переменных $x_i, i \in I$, доставляющий максимальное значение функции W и удовлетворяющий условиям (2), (3). Процесс определения этого набора представим как N-стадийный процесс, на каждом шаге которого $i = 1, 2, \ldots, N$; исходя из принципа оптимальности, выбирается значение $x_i = 0$ или $x_k = 1$. Стадии принятия решений отсчитываются в обратном порядке (справа налево) $1, 2, \ldots, i, \ldots, N$. Тогда стадия i означает, что до конца процесса принятия решения осталось i шагов.

Поскольку в данной задаче выбор на каждом шаге значения x_i, а следовательно, и его влияние на оптимизируемую функцию зависит от текущей грузоподъёмности рюкзака V_T, состояние системы определим последовательностью целых чисел $S = 0, 1, \ldots, j, \ldots, V$, интерпретируя их как распределение этой грузоподъёмности. На основании этого функцию Беллмана – как максимальное значение оптимизируемой функции от состояния системы – на каждом шаге $i = 1, 2, \ldots, N$ будем представлять функцией $B_i(S)$. Для первого шага N-стадийного процесса она будет иметь вид

$$B_1(S) = \max(c_1 \cdot x_1) \text{ по } x_1 = \begin{cases} 0, & \text{если } [S/a_1] < 1, \\ 1, & \text{если } [S/a_1] \geq 1, \end{cases} \quad (7)$$

где $[S/a_1]$ – целая часть числа S/a_1.
Таким образом, пока $[S/a_1] < 1$, $B_1(S) = 0$.

Для остальных $S = j, \ldots, V$, $B_1(S) = c_1$.

Для i-го шага $i = 2, \ldots, N$ функция Беллмана записывается так

$$B_i(S) = \max[c_i \cdot x_i + B_{i-1}(S - a_i \cdot x_i)]$$
$$\text{по } x_i = \begin{cases} 0, & \text{если } [S/a_i] < 1, \\ 0 \text{ или } 1, & \text{если } [S/a_i] \geq 1. \end{cases} \quad (8)$$

В свою очередь рекуррентное соотношение (8) можно интерпретировать следующим образом: пока $[S/a_i] < 1$, $B_i(S) = B_{i-1}(S)$; для $[S/a_i] \geq 1$, $B_i(S) = \max\{B_{i-1}(S), [c_i + B_{i-1}(S - a_i)]\}$.

Решение задачи о рюкзаке с использованием введённых соотношений выполняется традиционно. Вводится в рассмотрение две матрицы: $MB(i,s)_{N-1 \times V}$, $MX(i,s)_{N-1 \times V}$.

В матрицу MB построчно заносятся значения функции Беллмана, вычисляемые по выражениям (7), (8). В матрицу MX записываются оптимизирующие функцию Беллмана значения компонент вектора X. Максимальная стоимость предметов рюкзака вычисляется на основании компонент $(N-1)$-й строки матрицы MB по выражению

$$B_N(V) = \max\{B_{N-1}(V), [c_N + B_{N-1}(V - a_n)]\}.$$

При этом определяется и оптимальное значение $x_n^* = 0$ или 1. Остальные значения компонент вектора X вычисляются на основании значений элементов матрицы MX последовательно по выражениям

$$x_{n-1}^* = MX(n-1, V - a_n \cdot x_n^*),$$

$$x_{n-2}^* = MX(n-2, V - a_n \cdot x_n^* - a_{n-1} \cdot x_{n-1}^*), \ldots,$$

$$x_1^* = MX(1, V - a_n \cdot x_n^* - a_{n-1} \cdot x_{n-1}^* - \ldots - a_2 \cdot x_2^*).$$

В качестве эвристического рассматривается алгоритм, приближённо решающий оценочную задачу (4)–(6).

Он включает следующие действия.

Шаг 1. Установить n. Сформировать вектор «ценностей» единиц грузов

$$Y = \left(\frac{c_1}{a_1}, \frac{c_2}{a_2}, \ldots, \frac{c_n}{a_n} \right).$$

Шаг 2. Упорядочить компоненты вектора Y по убыванию и получить соответствующий вектор индексов предметов X.
Шаг 3. Положить начальный вес рюкзака $L = 0$, начальную «ценность» груза $z = 0$.
Шаг 4. Положить $i = 1$.
Шаг 5. Вычислить $L = L + A(x_i)$.
Шаг 6. Если $L \le V$, вычислить $z = z + c(x_i)$, иначе вычислить $L = L - c(x_i)$ и перейти к шагу 8.
Шаг 7. Положить $i = i + 1$; если $i \le n$, вернуться к шагу 5.
Шаг 8. Вычислить недогрузку рюкзака $\Delta V = V - L$ и остановиться.

Результаты эксперимента

Экспериментальные характеристики алгоритмов получены путём решения представительного набора случайных задач. Программы составлены на языке *Visual Basic*. Экспериментальные данные обрабатывались средствами пакета *Statgraphics* версии 5.1.

Всего каждым алгоритмом решалось 6 наборов случайных задач размером 10, 20, 30, 40, 50, 60 предметов по 100 задач в каждом наборе. Для алгоритмов, реализующих схему ветвей и границ и метод динамического программирования, определялись средние процессорные времена T_v, T_d счёта задач и недогрузки рюкзака R_v. Для эвристического алгоритма вычислялись средняя Δ_s и максимальная $\Delta_{\max}$ погрешности по отношению к точному решению задачи, стандартное отклонение S_t, 95-процентный доверительный интервал D_e, D_r для среднего Δ_s, а также средняя недогрузка рюкзака R_e.

Полученные данные приведены в табл. 1 и 2.

Таблица 1 Данные эксперимента для алгоритмов, реализующих схему ветвей и границ и метода динамического программирования

Количество предметов N	T_v	T_d	R_v
10	0,000547	0,000021	3,93
20	0,001641	0,000247	1,90
30	0,003867	0,000586	0,39
40	0,017031	0,001094	0,15
50	0,103281	0,002188	0,11
60	0,377344	0,003811	0,06

Таблица 2 Данные эксперимента для эвристического алгоритма

Количество предметов N	Δ_s	$\Delta_{\max}$	S_t	D_e	D_r	R_e
10	1,78	9,87	2,54	2,49	2,59	9,44
20	1,36	6,25	1,43	1,40	1,45	4,31
30	0,99	3,66	0,90	0,88	0,92	2,76
40	1,05	2,87	0,80	0,78	0,81	1,90
50	0,82	2,19	0,57	0,56	0,58	1,65
60	0,73	2,25	0,50	0,49	0,51	1,20

Данные таблиц показывают, что средняя Δ_s и максимальная $\Delta_{\max}$ погрешности эвристического алгоритма достаточно низки. Так, для размера задачи $N = 10$ они составляют меньше 2 и 10 процентов соответственно. Более того, с увеличением размера задачи N погрешности существенно убывают, так что, например, для $N = 60$ они составляют меньше 1 и 3 процентов. Наблюдается весьма малый разброс случайных S_t и узкий доверительный интервал $[D_e, D_r]$ для среднего, что свидетельствует о статистической устойчивости этих характеристик алгоритма.

Что касается временных характеристик, реализующих метод ветвей и границ и метод динамического программирования, то они отличаются. Эмпирические зависимости среднего времени счёта от размера задачи, полученные методом регрессивного анализа, приведены ниже. Для метода ветвей и границ характерна экспоненциальная зависимость $T_v = e^{-9,1212+0,133 \cdot N}$. Для метода динамического программирования – степенная – $T_d = e^{-17,0377} \cdot N^{2,804}$. Таким образом, по времени счёта более предпочтителен метод динамического программирования. Однако он требует существенного объёма памяти, который растёт как $2 \cdot [N \cdot (V+1)]$, т. е. примерно как $O(N^2)$. На практике возможно применение обоих методов, в том числе и эвристического, если требуется быстрое, прикидочное решение задачи.

Выводы

В данной статье авторы представили сконструированные и экспериментально исследованные алгоритмы, реализующие решения задачи о рюкзаке точным методом ветвей и границ, методом динамического программирования, а также эвристическим методом.

Практическая значимость исследований состоит в том, что получена оценка решения задач, оценка работы разработанных алгоритмов в среднем, что является важным фактором их применения.

Литература

1. Пападимитриу Х. Комбинаторная оптимизация. Алгоритмы и сложность / Х. Пападимитриу, К. Стайглиц. – М.: Мир, 1985. – 510 с.
2. Корбут А.А. Дискретное программирование. / А.А. Корбут, Ю.Ю. Финкельштейн. – М.: Наука, 1969. – 368 с.
3. Шкурба В.В. Задачи трёх станков / В.В. Шкурба. – М.: Наука, 1976. – 96 с.
4. Саати Т. Целочисленные методы оптимизации и связанные с ними экстремальные проблемы. / Т. Саати. – М.: Мир, 1973. – 304 с.
5. Казаков А.Я. Модифицированные модели Джейнса-Каммингса и квантовая версия задачи о рюкзаке / А.Я. Казаков // Журнал экспериментальной и теоретической физики. – 2003. – Т. 124, Вып. 6 (12). – С. 1264–1270.
6. Замкова Л.И. Булева двухкритериальная задача о рюкзаке / Л.И. Замкова // Известия Южного федерального университета. Технические науки. – 2009. – № 4. – С. 201 – 204.
7. Gilmore P.C. The Theory and Computation of Knapsack Functions / P.C. Gilmore, R.E. Gomory // Operations Research. – 1966. – Vol. 14, No. 6. – pp. 1045 – 1074.

References

1. Papadimitriu H., Stayglits K. *Kombinatornaya optimizatsiya. Algoritmyi i slozhnost* [Combinatorial optimization. Algorithms and Complexity]. Moscow, Mir Publ., 1985. 510 p.
2. Korbut A.A., Finkelshteyn Yu.Yu. *Diskretnoe programmirovanie* [Discrete programming]. Moscow, Nauka Publ., 1969. 368 p.
3. Shkurba V.V. *Zadachi trYoh stankov* [Task three machines]. Moscow, Nauka Publ., 1976. 96 p.
4. Saati T. *Tselochislennyie metodyi optimizatsii i svyazannyie s nimi ekstremalnyie problemyi* [Integer optimization techniques and associated extreme problems]. Moscow, Mir Publ., 1973. 304 p.
5. Kazakov A.Ya. Modifitsirovannyie modeli Dzheynsa-Kammingsa i kvantovaya versiya zadachi o ryukzake [Modified Jaynes-Cummings model and the quantum version of the knapsack problem]. *Zhurnal Eksperimentalnoy i Teoreticheskoy Fiziki*, 2003, Vol. 124, no. 6 (12). pp. 1264–1270.
6. Zamkova L.I. Buleva dvuhkriterialnaya zadacha o ryukzake [Boolean twocriterial knapsack problem]. *Izvestiya Yuzhnogo federalnogo universiteta. Tehnicheskie nauki*, 2009, no. 4. pp. 201–204.
7. Gilmore P.C., Gomory R.E. The Theory and Computation of Knapsack Functions. *Operations Research*. 1966, Vol. 14, no. 6. pp. 1045–1074.

Рецензент: П.Ф. Горбачёв, профессор, д.т.н., ХНАДУ.

Статья поступила в редакцию 16 февраля 2015 г.