М.Ш. Левин, А.В. Сафонов

Эвристический алгоритм для многокритериальной блочной задачи о рюкзаке

Аннотация. Предлагается эвристический аппроксимационный подход к решению многокритериальной блочной задаче о рюкзаке (multicriteria multiple choice problem). Приведены результаты численного эксперимента по сравнению предложенного алгоритма с точным переборным алгоритмом. Подробно рассматривается влияние параметров числа групп и числа элементов на эффективность использования эвристического алгоритма по сравнению с точным алгоритмом решения. Изучается влияние ресурсного ограничения на работу алгоритмов. В работе приведены результаты вычислительных экспериментов на различных наборах сгенерированных начальных данных и даны рекомендации по использованию предложенного аппроксимационного алгоритма в реальных задачах.

Ключевые слова: комбинаторная оптимизация, многокритериальная блочная задача о рюкзаке, эвристики, переборные алгоритмы, приближенные алгоритмы, динамическое программирование, вычислительный эксперимент.

Введение

Блочная задача о рюкзаке (multiple choice knapsack problem) [21, 31, 50] широко используется в различных областях, например: (1) планирование инвестиций (в частности, формирование инвестиционных «портфелей») [17, 18, 20, 52], (2) проектирование оверлейных структур программных комплексов или информационных систем [41], (3) выбор скорости передачи данных для многошаговых путей в коммуникационных сетях (transmission rate selection on a multi-hop path) [69], (4) размещение базовых станций в радио сетях [60], (5) перепроектирование коммуникационных сетей [45], (6) составление расписаний в системах GRID-вычислений [58], (7) планирование работы серверов в системах «клиент-сервер» [51], (8) планирование работы сложных радарных систем [54], (9) построение маркетинговых стратегий [43], (10) формирование планов технических проверок сложных систем [40], (11) формирование стратегий политиков [46], (12) проектирование программных и электронных систем (hardware) [65], (13) планирование перестройки (усиления) зданий [39], (14) композиция систем сервиса на базе Веб [5, 70], (15) организация диетического питания [68].

Блочная задача о рюкзаке [21, 31, 50] имеет достаточно простую и наглядную интерпретацию (выбор вариантов для частей некоторой системы с учетом общего ресурсного ограничения и максимизацией общей «полезности» выбранных вариантов) и соответствует простейшей версии задачи построения конфигурации системы [38]. Кроме того, задачи рассматриваемого типа часто являются составными частями более сложных задач комбинаторной оптимизации, например: общие задачи целочисленного программирования [19]), задачи назначения и размещения, в частности обобщенная задача о назначениях (GAP) [37], задачи составления расписаний [51]. Кроме базовой версии блочной задачи о рюкзаке широко используются различные ее версии [47], например: многорюкзачная постановка [42], многомерный (multidimensional) рюкзак [3, 24, 25, 61], стохастические постановки [7, 64], постановки с размытыми параметрами [36, 53].

Даже простейшая базовая постановка блочной задачи о рюкзаке является NP-трудной [21, 27, 28, 31, 50]. Список наиболее популярных методов решении задачи включает следующее: (а) переборные точные методы (метод ветвей и границ, метод динамического программирования и их

комбинации) [14-16, 24, 31, 32, 35], (б) приближенные алгоритмы с гарантированной погрешностью (по целевой функции, по ограничению (-ям)) [31, 41, 50], (в) генетические алгоритмы и многокритериальные эволюционные схемы решении (evolutionary multiobjective optimization) [2, 3, 9, 23, 29, 72], (г) эвристики/метаэвристики (например, Tabu search, Ant colony optimization) [4, 11, 12, 25, 55, 61, 66] и др.

Упомянем кратко несколько подходов. Достаточно часто применяется алгоритм «ветвей и границ» [14, 16, 22, 33, 49, 52, 63, 71], в котором необходимо сначала решить линеаризованную задачу с помощью «жадного» алгоритма, применив несколько критериев доминирования для отбрасывания «бесперспективных» вариантов, а затем решать редуцированную блочную задачу с использованием переборных схем для исправления некоторых переменных. При этом на каждом шаге требуется вычисление верхних границ, для чего необходимо решать линеаризованную блочную задачу с новыми начальными условиями. Существенную вычислительную сложность представлял первый этап – проверка на доминирование (в частности, LP-доминирование, описание [50, 57]), а также многократное повторение второго этапа. В результате алгоритмы имеют псевдополиномиальную сложность.

За псевдополиномиальное время блочную задачу о рюкзаке можно решить с использованием динамического программирования [1, 6, 8, 26, 68], представляющего собой перебор «в ширину» с применением критериев доминирования. Кроме того, были предложены псевдолинейные (для некоторых видов начальных данных) алгоритмы [48, 56, 57], основанные на поиске т.н. «ядра» (“core”) из «высокоперспективных» вариантов и расширении его при необходимости с использованием динамического программирования. Сначала с помощью алгоритма разбиения решается линеаризованная блочная задача и находится «ядро». Затем с помощью динамического программирования (метода “divide&conquer”) к «ядру» добавляются элементы из групп (одна за другой), заранее производя в них редуцирование по слабой верхней границе [10] и доминированию вариантов. Отметим, что все вышеприведенные схемы могут быть превращены в аппроксимационные схемы путем ослабления верхних границ на ε или прекращения перебора после заданного числа шагов. Однако, до настоящего времени они не были подробно рассмотрены. Табл. 1 содержит характеристики нескольких известных алгоритмов.

В данной работе предлагается эвристический алгоритм для многокритериальной блочной задачи о рюкзаке, позволяющий сократить вычислительную сложность алгоритмов, не использующих «ядро», за счет отказа от решения линеаризованной задачи с применением ресурсоемкого LP-доминирования и сокращения переборной фазы. При этом упорядочивание элементов по убыванию полезности и обычное доминирование используются. Предлагается сначала найти «перспективное» решение, удовлетворяющее ресурсному ограничению, путем последовательных переходов (замен элементов) от

Табл. 1. Обзор алгоритмов для решения блочной задачи о рюкзаке

Алгоритм	**Вычислительная сложность**	**Источник**
Динамическое программирование (ДП)	O(nc)	[68]
ДП + доминирование	O(n log max($\|N_i\|$)) + O(n'c)	[68]
ДП с редуцированием	O(min(2^{n+1},nc))	[68]
Ветвей-и-границ	O(n log max($\|N_i\|$)) + (*y ветвлений, y < n*)	
S&Z, Gl&Kl	y x O(n log r)	[22, 63]
Z	y x O(n)	[71]
D&W	y x O(r $\log^2$ (n/r))	[13]
Расширяющееся «ядро»	O(n) + r x (O($\|N_i\|$) + O($\|N_i'\|$ log $\|N_i'\|$) + O(m$\|N_i''\|$ log $\|N_i''\|$)), *m=1..r*	[56, 57]

наилучшего решения (выбранного без учета ресурсного ограничения) с минимальным уменьшением полезности или относительной полезности набора. После этого элементы «перспективного» решения заменяются элементами с меньшими ресурсными требованиями. Затем для полученного «ухудшенного» решения с помощью динамического программирования из ранее замененных элементов ищется оптимальный набор, максимально улучшающий решение, удовлетворяя при этом ресурсному ограничению. В целях подтверждения эффективности предложенного алгоритма проводится серия вычислительных экспериментов на сгенерированных наборах начальных данных.

Работа организована следующим образом: сначала формулируются рассматриваемые задачи комбинаторной оптимизации, затем описываются сравниваемые схемы решения, приводятся результаты и условия проведения вычислительных экспериментов (проведенные в среде Matlab [67]), наконец, проводится анализ результатов экспериментов.

1. Задачи комбинаторной оптимизации

1.1. Многокритериальное ранжирование

Рассмотрим альтернативы $A=\{A_1,...,A_j,...,A_n\}$, критерии $C=\{C_1,...,C_p,...,C_m\}$. Для каждой A_j задан вектор оценок $z_j=(z_{j1},...,z_{jp},...,z_{jm})$ по критериям $C_1,...,C_p,...,C_m$. Пусть μ_p - вес (важность) критерия C_p. Задача состоит в сравнении альтернатив (по векторам оценок z_j с учетом весов критериев μ_p) и формировании для каждой альтернативы порядковой оценки качества – приоритета. Описанная постановка относится к классу слабо структуризованных задач по классификации Г. Саймона [62]. В работе использована модификация метода порогов несравнимости Electre [59], предложенная в [44].

1.2. Блочная задача о рюкзаке

1.2.1. Бинарная задача

Пусть имеется n предметов: $1,...,n$. Рассмотрим n бинарных переменных $x=(x_1,...,x_i,...,x_n)$, где $x_i=1$ если предмет i выбран и 0 в обратном случае. Далее, если p_i – это полезность предмета i, w_i – его вес (т.е., требуемый ресурс) и c – это вместимость рюкзака (иными словами, общее ресурсное ограничение), наша задача состоит в том, чтобы найти бинарный вектор x, который максимизирует целевую функцию (функцию полезности) среди всех возможных векторов x, удовлетворяющих ресурсному ограничению. Теперь можно описать математически строго прототип задачи, известной как бинарная задача о рюкзаке [21, 31, 42, 50]:

Максимизировать $u=\sum_{i=1}^{n} p_i x_i$

При условии $\sum_{i=1}^{n} w_i x_i \le c$ и $x_i=0$ или 1, $i=1,\ldots,n$.

Похожие задачи возникают в прикладной математике, комбинаторике, теории сложности вычислений и криптографии. Также задача о рюкзаке возникает в тех случаях, когда имеет место распределение ресурсов с ограничениями. На практике чаще всего встречается приведенная выше задача о 0-1 рюкзаке (также называемая бинарной), где каждый предмет может быть или выбран и помещен целиком (1), или не включен в набор (0), что отличает ее от дробной задачи о рюкзаке. Бинарная задача о рюкзаке является частным случаем ограниченной задачи о рюкзаке (Bounded Knapsack problem), в которой существует ограниченное количество элементов каждого типа (в ее случае - только один предмет каждого типа). Отметим, что бинарная задача о рюкзаке является NP-трудной [21]. Это следует из того факта, что задача о сумме подмножества (частный случай рассматриваемой задачи при $p_i=w_i$) является NP-трудной, потому что она за полиномиальное время приводится к задаче о разбиении, являющейся базовой NP-трудной задачей, которую первоначально рассматривал Р. Карп [30].

1.2.2. Базовая версия блочной задачи

Данная задача является расширением задачи о рюкзаке [21,27,31,42]:

Максимизировать $u=\sum_{i=1}^{n} p_i x_i$

При условии $\sum_{i=1}^{n} w_i x_i \le c$

$$\sum_{i\in N_k} x_i = 1,\ k=1..r$$

$x_i = 0$ или 1, $i \in N = \{1,...,n\} = \bigcup_{k=1}^{r} N_k$

Предполагая, что $N_h \cap N_k = \varnothing, \forall h \neq k$

Эта задача является NP-трудной [21, 27, 28, 31, 71], что легко заметить: любая бинарная задача о рюкзаке, состоящая из r предметов с полезностями p_i и весами $w_i (i = 1,...,r)$, ресурсным ограничением *c*, эквивалентна следующей блочной задаче о рюкзаке, полученной заданием $n = 2r$, $p_i = w_i = 0$ для $i = r+1,...,2r$ и $N_k = \{k, r+k\}$ для $k = 1,...,r$. Необходимо отметить, что сложность блочной задачи о рюкзаке может быть во многих случаях значительно уменьшена с помощью применения фазы редукции, которая основана на отношениях доминирования между предметами. Мы можем сформулировать следующий тривиальный критерий доминирования: для того, чтобы предмет $i \in N_k$ являлся доминируемым (т.е. $x_i = 0$) необходимо и достаточно, чтобы для некоторого предмета $j \in N_k$ выполнялось $p_i \le p_j$ и $w_i \ge w_j$ (Рис. 1).

1.2.3. Многокритериальная блочная задача

Пусть вместо полезности элемента p_i используется вектор из *m* компонентов: $(p_i^1,..., p_i^j,..., p_i^m)$ [34, 38, 45, 65]. Оптимизация (поиск эффективных по Парето решений) осуществляется по векторной целевой функции:

$$\left(\sum_{i=1}^{n} p_i^1 x_i,..., \sum_{i=1}^{n} p_i^j x_i,..., \sum_{i=1}^{n} p_i^m x_i \right)$$

При условии $\sum_{i=1}^{n} w_i x_i \le c$

$$\sum_{i \in N_k} x_i = 1,\ k = 1..r$$

$$x_i = 0 \text{ или } 1,\ i \in N = \{1,...,n\} = \bigcup_{k=1}^{r} N_k$$

Предполагая $N_h \cap N_k = \varnothing, \forall h \neq k$

Поиск решения с использованием метода Парето-эффективных точек не приводит к ожидаемому результату, когда существует множество Парето-несравнимых точек. В этих случаях метод порогов несравнимости Электре [59] может предоставить лучшие результаты, т.к. в его основе не лежат отношения полного доминирования среди альтернатив. Поэтому используется предварительное сведение многокритериальной задачи на основе [45] многокритериального ранжирования элементов $(p_i^1,..., p_i^j,..., p_i^m)$ для получения приоритета t_i для каждого элемента (в диапазоне $[1; +\infty)$, приоритет 1 является худшим). Таким образом, получается обычная блочная задача о рюкзаке.

Необходимо отметить, что при использовании данного метода необходимо обратить внимание на диапазоны оценок по критериям в разных группах. За счет масштабирующих множителей можно добиться больших приоритетов у элементов тех групп, которые являются более значимыми. Например, пусть имеется 6 групп, среди них одна – высокой важности, три – средней важности, две – низкой важности. В этом случае можно использовать множители {2;1;1;1/2;1/2}. Далее рассмотрим несколько возможных схем решения блочной задачи о рюкзаке.

2. Схемы решения блочной задачи

2.1. Схема динамического программирования

Схема динамического программирования представляет собой перебор «в ширину» с применением критериев доминирования. Базовым является следующий рекурсивный алгоритм, имеющий псевдополиномиальную сложность. Рассмотрим подзадачу (блочной задачи о рюкзаке), состоящую из подмножеств $N_1,\ldots, N_j$ ($1 \le j \le r$), причем вместимость равна $\hat{c}$ ($0 \le \hat{c} \le c$). Пусть $F(j, \hat{c})$ - рекурсивная функция, которая дает значение полезности оптимального решения:

$$F(j,\hat{c}) = \left\{ \begin{array}{l} -\infty,\ \hat{c} = 0,..., \sum_{k=1}^{j} \tilde{w} - 1 \\ \max\left(p_l + F(j-1, \hat{c} - w_l) : l \in N_j, w_l \le \hat{c}\right), \hat{c} = \sum_{k=1}^{j} \tilde{w}_k - 1 \end{array} \right\}$$

Причем $\tilde{w}_k = \min(w_l : l \in N_k)$ для $k = 1..r$, а $F(j,\hat{c}) = -\infty$, если подзадача не имеет подходящего решения (не нарушающего ресурсное ограничение $\hat{c}$). Решение может быть найдено путем вычисления $F(r,c)$. Для этого необходимо пройти *r* этапов (увеличивая *j* от 1 до *r*), увеличивая $\hat{c}$ от 0 до *c* и просматривая все предметы из *j*-го подмножества. Таким обра-

зом, это решение потребует $O(nc)$ времени. Данная схема соответствует алгоритмам динамического программирования, предложенным в первых работах [6, 8]. Для более эффективной реализации – перехода от рекурсии к циклу – схема была незначительно модифицирована, подробно это описано в [50, 68]. Кроме того, на предварительном этапе проводится редуцирование задачи – отбрасывание доминируемых элементов, что потребует $O(n \log(\max_{k=1..r} N_k))$ времени. Рис. 1 иллюстрирует процесс отбрасывания доминируемых элементов.

2.2. Эвристический алгоритм

Алгоритм включает следующие шаги:

Шаг 1. Упорядочивание элементов по убыванию приоритета – получаем единый Список из n элементов.

Шаг 2. Отбрасывание доминируемых элементов (критерий доминирования – см. 2.2.2) внутри каждой из r групп.

Шаг 3. Заполнение «рюкзака» лучшими элементами из каждой группы, не учитывая ресурсное ограничение. Если полученный набор удовлетворяет ресурсному ограничению – решение найдено.

Шаг 4. Вычисление матриц градиентов: относительного уменьшения приоритета ($\Delta t_i / \Delta w_i$) и абсолютного уменьшения приоритета (Δt_i) при проходе по Списку, причем разным группам соответствуют разные строки матриц. Размерность матриц определяется как $r*(max\{N_i\}-1)$. Если в группе j всего h элементов, а наибольшее число элементов среди групп s ($s > h$), то в матрицах градиентов в строке j последние (s-h-1) ячеек будут заполнены 0.

В дальнейшем использованию матрицы градиента относительного уменьшения приоритета соответствуют шаги x.1, а абсолютного – шаги x.2 соответственно.

Шаг 5.1 и 5.2. По матрицам градиента находится элемент, замена на который обеспечит наименьшую потерю приоритета набора. Производится замена. Шаг повторяется, пока не будет удовлетворено ресурсное ограничение. Запоминаются замененные элементы. С этими элементами ассоциируются уменьшение ресурсного требования Δw и уменьшение приоритета набора Δt. Причем, если в группе j уже было проведено (k-1) замен, то, при проведении k-й замены, (k-1) замененных элементов получат приращения Δw и Δt соответственно. Таким образом, Δw и Δt для замененных элементов показывают изменения соответствующих параметров относительно актуального решения.

Шаг 6.1 и 6.2. «Ухудшающая замена». Все выбранные элементы (по 1 из каждой группы) заменяются на элементы с более низкими приоритетом и весом (из тех же групп соответственно). Запоминаются замененные элементы.

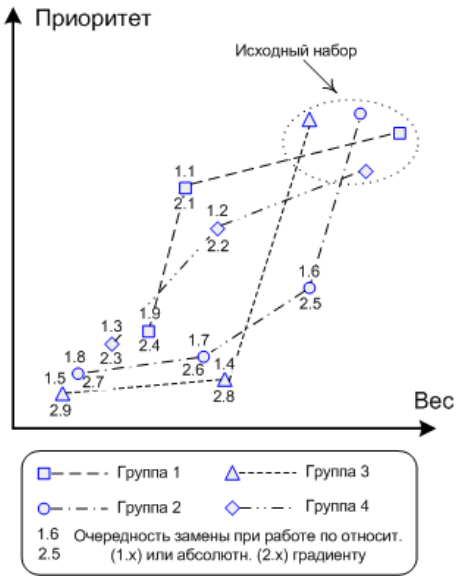

Рис. 1. Иллюстрация схемы работы алгоритма на шаге 5.1 и 5.2

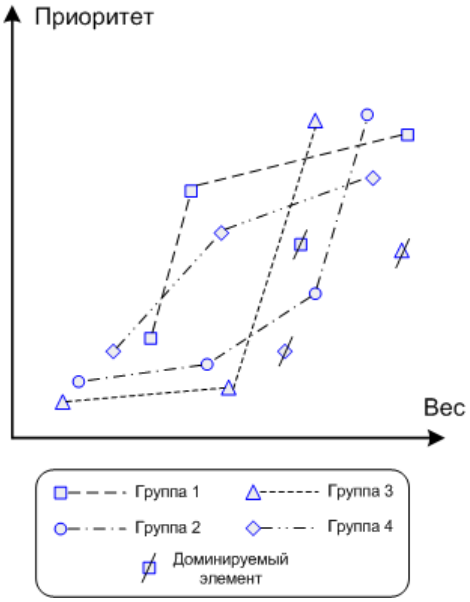

Рис. 2. Доминирование элементов

Шаг 7.1 и 7.2. На всех ранее замененных элементах с использованием алгоритма ветвей и границ решается задача поиска удовлетворяющего ограничению по доступному ресурсу набора-замены, который заменяет 1,2,… или все r элементов, максимально улучшая набор. Таким образом, решается модифицированная (без требования выбора элементов из каждой группы) блочная задача о рюкзаке, причем приоритетами и весами элементов являются их уменьшения при выборе данных элементов на замену (см. шаг. 5), а ресурсное ограничение определяется как исходный ресурс за вычетом требуемого для набора, полученного на шаге 6.

Шаг 8. Из наборов, полученных в 7.1 и 7.2, выбирается лучший (с большим приоритетом) – решение найдено.

Рис. 2 содержит иллюстративный материал по работе алгоритма. Блок-схема эвристического алгоритма приведена на Рис. 3.

3. Вычислительные эксперименты

Представленные выше (п.п.3.1-3.2) алгоритмы были реализованы в среде Matlab 6.5 [67]. Схема вычислительного эксперимента иллюстрируется на Рис. 4. Затраченное время определялось с использованием встроенных команд *tic* и *toc*. Расчеты проводились на ПК следующей конфигурации: AMD Athlon64 X2 2ГГц, 1,5 Гбайт ОЗУ. Для вычислений использовалось одно из двух ядер.

Переменные вычислительного эксперимента представлены в Табл. 2. Вычисления проводились для задачи различных масштабов: менялось число групп r и общее число элементов n. Кроме того, изменялся параметр λ строгости ресурсного ограничения, которое определяется как $\lambda \cdot \sum_{j=1}^{r} \min_{i \in N_j} w_i$. Для $\lambda \geq 1$ решение заведомо существует. Для всех наборов исходных данных дополнительно подтверждается, что $\lambda \cdot \sum_{j=1}^{r} \min_{i \in N_j} w_i < \frac{1}{2} \sum_{j=1}^{r} \max_{i \in N_j} w_i$. По данному условию определено, что $\lambda \in (1;3]$ – в этом случае ограничение нарушается не более чем в 10% сгенерированных наборов. А при $\lambda \geq 3,5$ более чем 50% наборов начальных данных не удовлетворяли сформулированному выше условию.

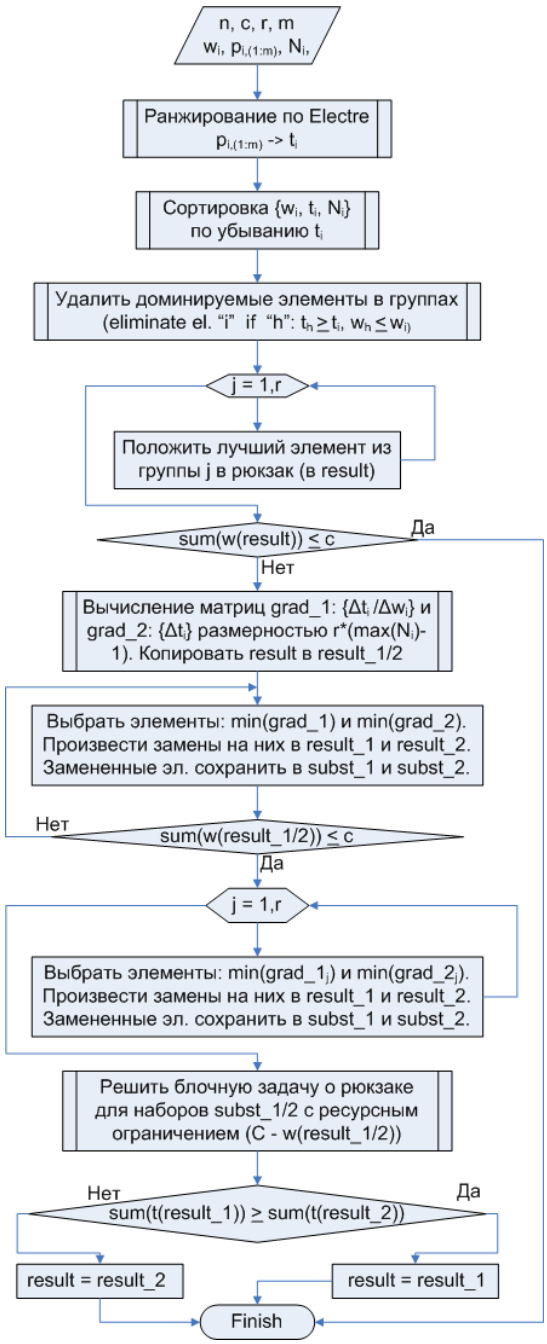

Рис. 3. Блок-схема эвристического алгоритма

Использовался генератор начальных данных в следующей конфигурации:

1. Задание разбиения на группы $N = \bigcup_{k=1}^{r} N_k = \{1,...,n\}$ на основе случайных величин со стандартным нормальным распределением (*randn*(1,1)). Если не удовлетворено дополнительное условие – не менее 2 элементов в группе – генерируется заново.

Рис. 4. Общая схема вычислительного эксперимента

2. Задание оценок (целочисленных) по критериям из диапазона [1,10] с использованием матрицы равномерно распределенных случайных величин.

3. Задание значений средних ресурсных требований для групп с помощью случайных величин со стандартным нормальным распределением (*randn*(1,1)).

4. Определение ресурсных требований каждого элемента как функции-произведения от (i) суммы оценок элемента по критериям, (ii) среднего ресурсного требования группы и (iii) случайного множителя вида $\rho + \gamma \cdot randn(1,1)$.

Параметры генератора подбирались таким образом, чтобы степень коррелированности веса и приоритета у элементов соответствовали реальным задачам. Использование линеаризации через сумму оценок по критериям, а не метода Электре позволяет (наравне с внесением случайного поправочного члена) отойти от чрезмерно коррелированных наборов исходных данных, которые приводят к вырождению/преобразованию рассматриваемой задачи к классу задач о разбиении (subset-sum problem), в которых $t_i = w_i$ для всех i. Из условия приближения модельных расчетов к рассматриваемым нами реальным задачам [39] также выбиралось и число групп, на которое делится весь набор, т.е. среднее число элементов в группе. Наконец отметим, что для решения использовалось сведение многокритериальной блочной задачи о рюкзаке к одному параметру оценки с использованием ранжирования по Электре с пороговыми коэффициентами $\alpha = 0,7$ и $\beta = 0,3$. Для всех комбинаций n,r,λ проводились вычисления для 20 сгенерированных наборов начальных данных. Для $n = 100$, $r = 12$ проводились вычисления только для 5 наборов из-за большого времени работы точного алгоритма.

Примечание. При $\lambda = 1$ аппроксимационная схема превращается в схему динамического программирования с несколько большим временем выполнения из-за дополнительных операций, при данном λ не нужных. Впрочем, при таком λ отсутствует смысл решения задачи оптимизации, т.к. имеется лишь одно допустимое решение.

Табл. 2. Переменные вычислительного эксперимента

Параметр	**Описание**	**Характер**
n	Общее число элементов	Начальные данные
r	Число групп	
λ	Параметр строгости ресурсного ограничения	
$\varphi_{ср}$	Средняя оценка эвристического решения, в долях от точного решения (ДП)	Результаты
ρ	Доля случаев совпадения эвристического решения с точным решением (ДП)	
$\varphi_{мин}$	Минимальная оценка эвристического решения, в долях от точного решения (ДП)	
$T_{эвр.}$	Среднее затраченное время, эвристика	
$T_{дп}$	Среднее затраченное время, точная схема (ДП)	
θ	Среднее затраченное время для эвристики в долях от времени точной схемы (ДП)	

4. Обсуждение результатов вычислений

Анализ приведенных в Табл. 3 результатов приводит к выводу о том, что предложенная схема позволяет получать решения, близкие к точному, затрачивая при этом небольшое время даже для значительного числа элементов и групп благодаря отличной масштабируемости относительно числа групп. При этом получаемая погрешность ниже, чем погрешность экспертной оценки – предполагалась оценка по десятибалльной шкале, соответственно, ε = 10%, что меньше чем наихудшая полученная ошибка для $n \geq 90$ и $r \geq 7$. Для меньших значений n и r использование аппроксимационной схемы менее целесообразно, для больших же – может быть рекомендовано, особенно с учетом увеличения точности при увеличении числа групп, что является следствием использования матриц градиентов.

При увеличении числа групп наблюдается улучшение качества получаемого решения – растут все оцениваемые параметры: средняя оценка, минимальная (худшая) оценка, доля совпадений с точным (Рис. 5-8). В то же время, увеличение числа элементов при сохранении числа групп приводит к ухудшению получаемого решения (Рис. 9-10).

Отдельно необходимо отметить существенное влияние строгости ресурсного ограничения (параметра λ) на эффективность предложенного алгоритма, в частности, при ослаблении ресурсного ограничения наблюдается как рост точности (Рис. 5-10), так и ускорение работы (Рис. 11-12). В то же время, проведенные оценки показывают, что предложенный алгоритм недостаточно эффективен при очень жестких ресурсных ограничениях (параметр строгости λ стремится к 1).

Заключение

В работе был предложен и исследован эвристический алгоритм решения блочной задачи о рюкзаке. Было проведено сравнение эффективности предложенного алгоритма с точным и исследовано влияние параметров числа элементов и групп, а также строгости ресурсного ограниче-

Табл. 3. Сравнение эвристического алгоритма с точным по качеству получаемого решения и затраченному времени

N	r	λ	$\varphi_{ср}$	ρ	$\varphi_{мин}$	$T_{эвр.}$, с	$T_{дп}$, с	θ
80	6	1,5	0,9795	0,50	0,8993	0,42	3,14	0,27987
90	7	1,5	0,9923	0,75	0,9590	1,49	25,57	0,13529
80	8	1,5	0,9936	0,65	0,9797	2,35	52,38	0,16589
80	10	1,5	0,9979	0,70	0,9803	8,39	826,93	0,03612
100	8	1,5	0,9865	0,45	0,9460	9,05	522,82	0,10037
80	12	1,5	0,9974	0,80	0,9837	23,01	1329,09	0,01723
100	12	1,5	0,9885	0,80	0,9523	15,23	17292,34	0,00429
80	6	2,0	0,9891	0,70	0,9592	0,18	5,96	0,03734
90	7	2,0	0,9917	0,70	0,9411	0,36	44,59	0,01777
80	8	2,0	0,9993	0,90	0,9897	0,65	101,99	0,00999
80	10	2,0	0,9997	0,95	0,9899	1,20	910,83	0,00389
100	8	2,0	0,9986	0,75	0,9789	0,96	484,39	0,00342
80	12	2,0	1,0000	1,00	1,0000	2,61	1845,77	0,00069
100	12	2,0	0,9990	0,90	0,9881	6,13	16466,30	0,00020
80	6	3,0	1,0000	1,00	1,0000	0,03	4,50	0,00911
90	7	3,0	1,0000	1,00	1,0000	0,05	71,29	0,00113
80	8	3,0	1,0000	1,00	1,0000	0,04	92,63	0,00107
80	10	3,0	1,0000	1,00	1,0000	0,11	902,77	0,00062
100	8	3,0	1,0000	1,00	1,0000	0,07	299,10	0,00048
80	12	3,0	1,0000	1,00	1,0000	0,23	1928,88	0,00041
100	12	3,0	1,0000	1,00	1,0000	0,21	17434,56	0,00002

ния. Численные эксперименты подтвердили (относительную) быстроту выполнения предложенного аппроксимационного алгоритма, при этом получаемая погрешность была незначительна (для рассматриваемых параметров задачи).

Благодаря скорости выполнения можно проводить серии вычислений для нескольких вариантов оценки элементов по критериям – этому может соответствовать оценка несколькими экспертами или различные начальные условия/требования. Оптимальное решение в данном случае необходимо искать по пересечению набора решений. Кроме того, можно проводить серии вычислений для нескольких ресурсных ограничений, что позволит пронаблюдать поведение изучаемой системы в динамике.

В качестве перспективного направления исследований можно выделить разработку версии предложенного эвристического алгоритма, работающую начиная не с лучшего решения, а с худшего, что позволит более эффективно решать рассматриваемые задачи в условиях очень строгого ресурсного ограничения. В дополнение к этому, может быть выработан критерий предпочтения той или иной версии алгоритма в зависимости от строгости ресурсного

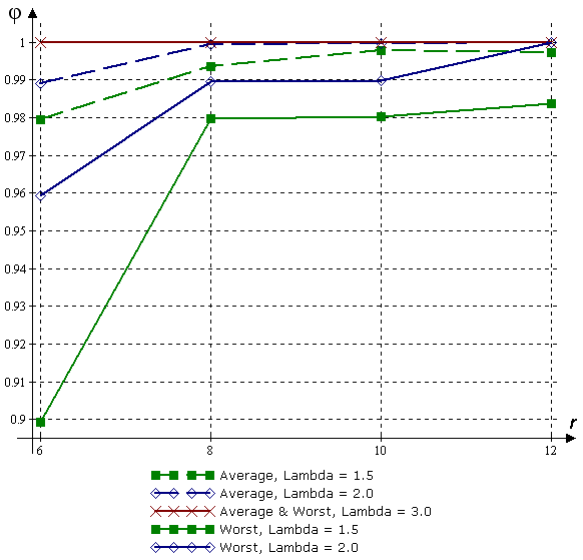

Рис. 5. Средние и минимальные оценки в долях от точного решения, n=80, r={6;8;10;12}.

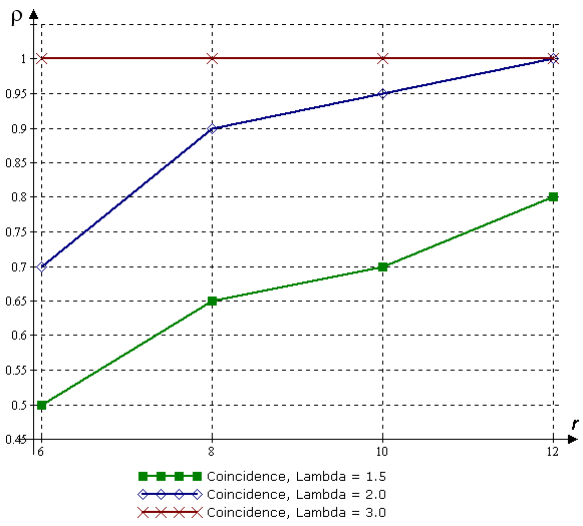

Рис. 6. Доля случаев совпадения с точным решением, n=80, r={6;8;10;12}.

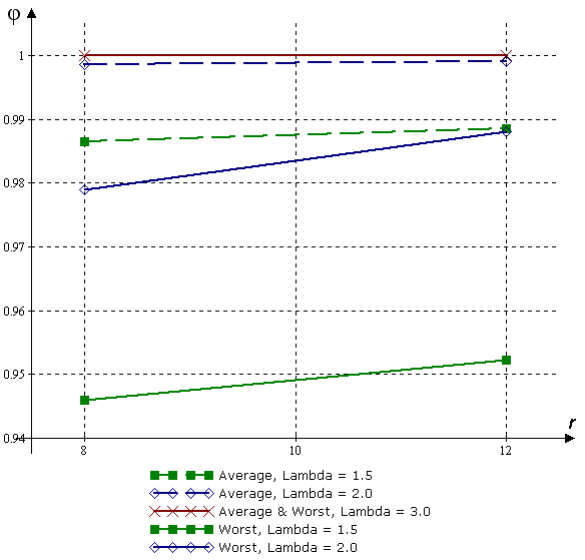

Рис. 7. Средние и минимальные оценки в долях от точного решения, n=100, r={8;12}.

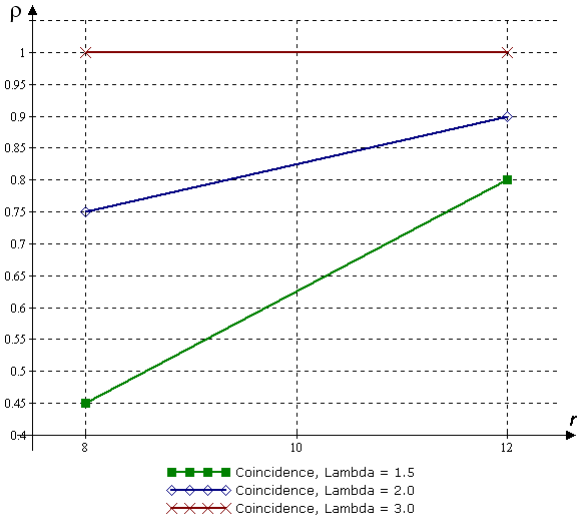

Рис. 8. Доля случаев совпадения с точным решением, n=100, r={8;12}.

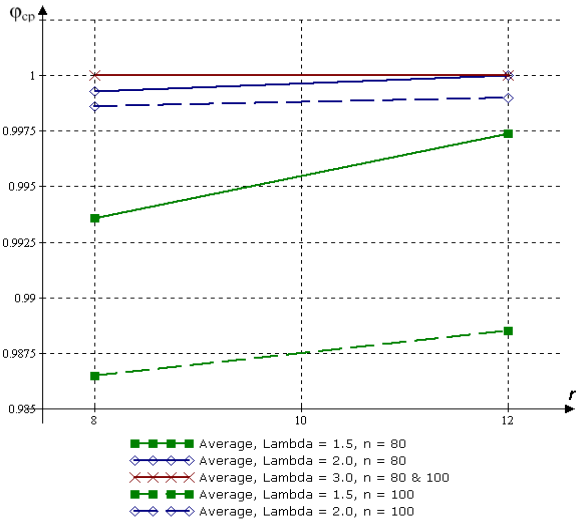

Рис. 9. Средние оценки в долях от точного решения, n={80;100}, r={8;12}

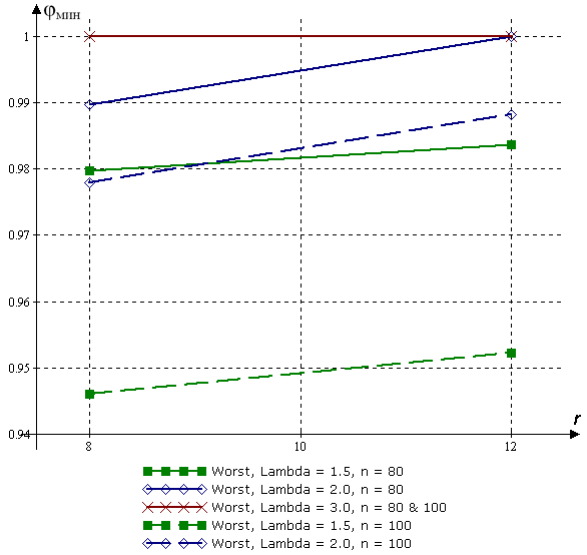

Рис. 10. Минимальные оценки в долях от точного решения, n={80;100}, r={8;12}

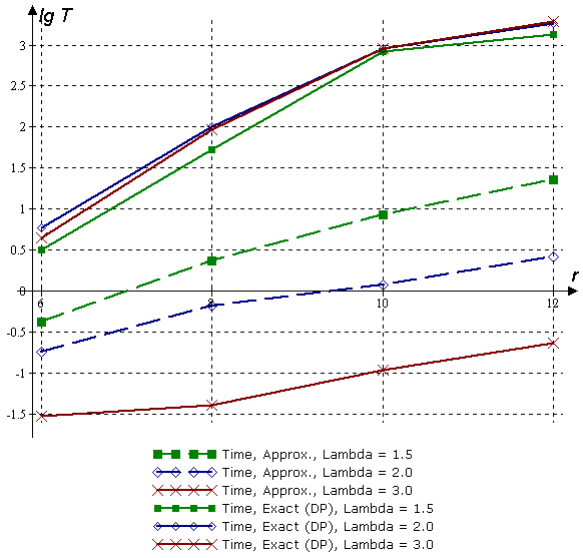

Рис. 11. Среднее затраченное время (lg) для точной и аппроксимац. схем, n=80, r={6;8;10;12}

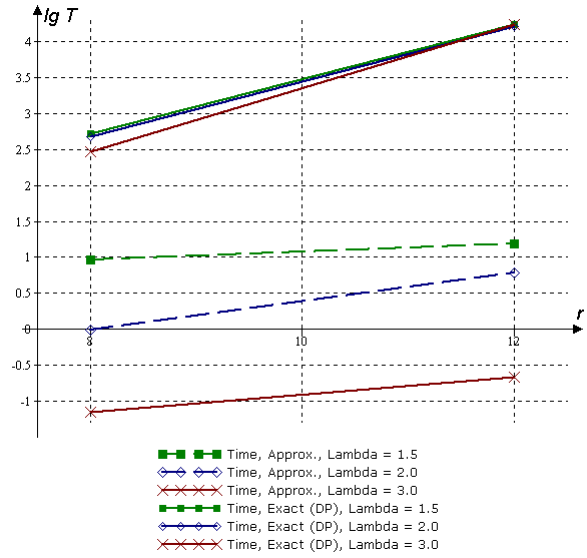

Рис. 12. Среднее затраченное время (lg) для точной и аппроксимац. схем, n=100, r={8;12}

ограничения (например, параметра λ). Также имеет смысл исследовать возможность использования различной глубины «ухудшающей замены» в зависимости от параметров задачи (например, числа элементов в группе) – т.к. используемая в настоящей версии глубина в один шаг может быть недостаточной при условии большего числа элементов в группе. Кроме того, эвристический алгоритм может быть доработан для эффективного проведения расчетов для нескольких ресурсных ограничений сразу. Перспективным направлением выглядит и адаптация алгоритма для многопроцессорных сред. Наконец, следует отметить, что представляется целесообразным в дальнейшем расширить число сравниваемых алгоритмов (например, за счет генетических алгоритмов).

Литература

1. Ahrens, J.H., Finke G. Merging and sorting applied to the zero-one knapsack problem. Oper. Res., 1975, 23(6), pp. 1099-1109.
2. Aickelin U. Genetic Algorithms for Multiple-Choice Optimization Problems. PhD Thesis, Univ. of Wales Swansea, 1999.

3. Akbar M.M., Rahman M.S., Kakobad M., Manning E.G., Shoja G.C. Solving the Multidimensional Multiple-choice Knapsack Problem by constructing convex hulls. Comp. and Oper. Res., 2006, 33(5), pp. 1259-1273.
4. Alves M.J., Climaco J. An interactive method for 0-1 multiobjective problems using simulated annealing and tabu search. J. of Heuristics, 2000, 6(3), pp. 385-403.
5. Ardanga D., Pernici B. Global and local QoS guarantee in Web service selection. In: C Bussler et al. (Eds.), BPM 2005 Workshops, LNCS 3812, Springer, 2006, pp. 32-46.
6. Bellman R. Dynamic Programming, Princeton, NJ: Princeton University Press, 1957.
7. Benoist T., Bourreau E., Caseau Y., Rottembourg B. Towards stochastic constraint programming: A study of online multi-choice knapsack with deadlines. In: Proc. of 7th It. Conf. "Principles and Practice of Constraint Programming CP 2001", LNCS 2239, Springer, Cyprus, 2001, pp. 61-76.
8. Dantzig G.B. Discrete variable extremum problems. Oper. Res., 1957, 5(2), pp. 266-288.
9. Deb K. Multi-Objective Optimization using Evolutionary Algorithms. Wiley, New York, 2001.
10. Dembo R.S., Hammer P.L. A reduction algorithm for knapsack problems. Methods of Oper. Res., 1980, 36, pp. 49-60.
11. Dorigo M., Blum C. Ant colony optimization theory: A survey. Theoretical Computer Science, 2005, 344(2-3), pp. 243-278.
12. Dorigo M., Stutzle T. The ant colony optimization metaheuristics: Algorithms, applications, and advances. In: F. Glover, G.A. Kochenberger, Handbook of Metaheuristics, Springer, 2006, pp. 250-285.
13. Dudzinski K., Walukiewicz S. A fast algorithm for the linear multiple-choice knapsack problem. Oper. Res. Lett., 1984, 3(4), pp. 205-209.
14. Dudzinski K., Walukiewicz S. Exact methods for the knapsack problem and its generalizations. Eur. J. of Oper. Res., 1987, 28(1), pp. 3-21.
15. Dyer M.E., Kayal N., Walker J. A branch-and-bound algorithm for solving the multiple choice knapsack problem. J. of Comput. and Appl. Math., 1984, 11(2), pp. 231-249.
16. Dyer M.E., Riha W.O., Walker J. A hybrid dynamic programming/branch-and-bound algorithm for the multiple choice knapsack problem. J. of Comput. and Appl. Math., 1995, 58(1), pp. 43-54.
17. Ehrgott M., Klamroth K., Schwem C. An MCDM approach to portfolio optimization. Eur. J. of Operational Research, 2004, 155(3), pp. 752-770.
18. Elton E.J., Gruber M., Brown S.J., Goetzmann W.N., Modern Portfolio Theory and Investment Analysis. 6 ed., Wiley, New York, 2002.
19. Fisher M.L. The Lagrangian relaxation method for solving integer programming problems. Management Science, 1981, 27(1), pp. 1-18.
20. Fiterman A.E.,Timkovsky V.G. Basket problems in margin calculation: Modeling and algorithms. Eur. J. of Oper. Res., 2001, 129(1), pp. 209-223.
21. Garey M.R., Johnson D.S. Computers and Intractability: The Guide to the Theory of NP-Completeness. San Francisco: W.H.Freeman and Company, 1979.
22. Glover F., Klingman D. A O(n log n) algorithm for LP knapsacks with GUB constraints. Mathematical Programming, 1979, 17(1-3), pp. 345-361.
23. Hadj-Alouane A.B., Bean J.C. A genetic algorithm for the multiple-choice integer program. Oper. Res., 1997, 45(1), pp. 92-101.
24. Han B., Leblet J., Simon G. Hard multidimensional multiple choice knapsack problems: an empirical study. Comp. and Oper. Res, 2010, 37(1), pp. 172-181.
25. Hifi M., Michrafy M., Sbihi A., Heuristc algorithms for the multiple-choice multidimensional knapsack problem. J. of Oper. Res. Soc., 2004, 55(12), pp. 1323-1332.
26. Horowitz E., Sahni S. Computing partitions with applications to the Knapsack Problem. J. of the ACM, 1974, 21(2), pp. 277-292.
27. Ibaraki T. Enumerative Approaches to Combinatorial Optimization, Part 1. Annals of Operations Research, 10(1), Basel: J.C. Baltzer, 1987.
28. Ibaraki T. Enumerative Approaches to Combinatorial Optimization, Part 2. Annals of Operations Research, 11(1), Basel: J.C. Baltzer, 1987.
29. Jaszkiewicz A. On the computational efficiency of multiple objective metaheuristics. The knapsack problem case study. Eur. J.of Oper. Res., 2004, 158(2), pp. 418-433.
30. Karp R.M. Reducibility among combinatorial problems. In: R.E. Miller, J.W. Thatcher (Eds.), Complexity of Computer Computations, Plenum Press, New York, 1972, pp. 85-103.
31. Kellerer H., Pferschy U., Pisinger D. Knapsack Problems, Berlin: Springer, 2004.
32. Klamroth K., Wiecek M.M. Dynamic programming approaches to the multiple criteria knapsack problem. Naval Research Logistics, 2000, 47(1), pp. 5-76.
33. Kolesar P.J. A branch and bound algorithm for the knapsack problem. Management Science, 1967, 13(9), pp. 723-735.
34. Kozanidis G. Solving the linear multiple choice problem with two objectives: Profit and equity. Computational Optimization and Applications, 2009, 43(2), pp. 261-294.
35. Kozanidis G., Melachrinoudis E. A branch and bound algorithm for 0-1 mixed integer knapsack problem with linear multiple choice constraints. Comp. and Oper. Res., 2004, 31(5), pp. 695-711.
36. Kuchta D. A generalization of an algorithm solving the fuzzy multiple choice knapsack problem. Fuzzy Sets and Systems, 2002, 127(2), pp. 131-140.
37. Laporte G. The vehicle routing problem: An overview of exact and approximate algorithms. Eur. J. of Oper. Res., 1992, 59(2), pp. 345-358.
38. Levin M.Sh. Combinatorial optimization in system configuration design. Autom. and Remote Control, 2009, 70(3), pp. 519-561.
39. Levin M.Sh., Danieli M.A. Hierarchical decision making framework for evaluation and improvement of composite systems. Informatica, 2005, 16(2), pp. 213-240.
40. Levin M.Sh., Merzlyakov A.O. Composite combinatorial scheme of test planning. IEEE Region 8 Int. Conf. "Sibircon-2008", Novosibirsk, 2008, pp. 291-295.
41. Левин М.Ш. Одна экстремальная задача организации данных // Изв. АН СССР, сер. Техн. Кибернетика. 1981. 5, С. 103-112.
42. Левин М.Ш. Комбинаторные оптимизационные модели рюкзачного типа // Измерение, контроль, автоматизация. 1988. 4(68), С. 51-63.
43. Левин М.Ш. Комбинированная схема для формирования стратегии маркетинга // Бизнес Информатика. 2009. 2, С. 42-51.

44. Левин М.Ш., Михайлов А.А. Фрагменты технологии стратификации множества объектов: Препринт. -М.: ВНИИСИ, 1988.
45. Левин М.Ш., Сафонов А.В. Проектирование и перепроектирование конфигурации оборудования коммуникационной сети // Информационные технологии и вычислительные системы. 2006. 4, С. 63-73.
46. Левин М.Ш., Фимин А.В. Комбинаторная схема анализа политических кандидатов и их стратегий // Информационные системы. 2009. 9(2), С. 83-92.
47. Lin E.Y.-H. A bibliographical survey on some well-known non-standard knapsack problems. INFOR, 1998, 36(4), pp. 274-317.
48. Martello S., Toth P. A new algorithm for the 0-1 knapsack problem. Management Science, 1988, 34(5), pp. 633-644.
49. Martello S., Toth P. Upper-bounds and Algorithms for Hard 0-1 Knapsack Problems. Research Report DEIS, University of Bologna, OR/93/04.
50. Martello S., Toth P. Knapsack Problem: Algorithms and Computer Implementation. Wiley, New York, 1990.
51. Mejia-Alvarez P., Levner E., Mosse D. Adaptive scheduling server for power-aware real-time tasks. ACM Trans. on Embedded Computing Systems, 2004, 3(2), pp. 287-306.
52. Nauss R.M. The 0-1 knapsack problem with multiple choice constraints. Eur. J. of Oper. Res., 1978, 2(2), pp. 125-131.
53. Okada S., Gen M. Fuzzy multiple choice problem. Fuzzy Sets and Systems. 1994, 67(1), pp. 71-80.
54. Orman A.J., Modelling for the control of a complex radar system. Comp. and Oper. Res., 1998, 25(3), pp. 239-249.
55. Osman I., Kelly J.P. (Eds.). Meta-Heuristics: Theory and Applications. Springer, 1996.
56. Pisinger D. A minimal algorithm for the multiple-choice knapsack problem. Eur. J. of Oper. Res., 1995, 83(2), pp 394-410.
57. Pisinger D. An expanding-core algorithm for the exact knapsack problem. Eur. J. of Oper. Res., 1995, 87(1), pp. 175-187.
58. Prodan R., Wieczorek M. Bi-Criteria scheduling of scientific grid workflow. IEEE Trans. on Automation Science and Engineering, 2009 (article in press).
59. Roy B. Multicriteria Methodology for Decision Aiding. Kluwer, 1996.
60. Shabany M., Navaie K. Joint pilot power adjustment and base station assignment for data traffic in cellular CDMA networks. 2004 IEEE/Sarnoff Symp. on Advances in Wired and Wireless Communication, 2004, pp. 179-183.
61. Shahriar A.Z., Akbar M.M., Rahman M.S., Newton M.M. A multiprocessor based heuristic for multi-dimensional multiple-choice knapsack problem. The J. of Supercomputing, 2008, 43(3), pp. 257-280.
62. Simon H.A., Newell A. Heuristic problem solving: The next advance in operations research, Oper. Res., 1958, 6(1), pp. 1-10.
63. Sinha P., Zoltners A.A. The multiple-choice knapsack problem. Oper. Res., 1979, 27(3), pp. 503-515.
64. Snyder L.V., Daskin M.S. Stochastic p-robust location problem. IIE Transaction, 2006, 38(11), pp. 971-985.
65. Sousa J.P., Poladian V., Garlan D., Schmerl B., Shaw M. Task-based adaptation for ubiquitous computing, IEEE Trans. on SMC, Part C, 2006, 36(3), pp. 328-340.
66. Teghem J., Tuyttens D., Ulungu E.L. An interactive heuristic method for multi-objective combinatorial optimization. Comp. and Oper. Res., 2000, 27(7-8), pp. 621-634.
67. The MathWorks website http://www.mathworks.com/
68. Toth P. Dynamic programming algorithms for the zero-one knapsack problem. Computing, 1980, 25(1), pp. 29-45.
69. Wang Q., Xu Y., Xu C., Chen G. Energy efficiency rate selection on a path in IEEE 208.11-based multi-hop network. J. of Inform. and Comput. Sci., 2007, 4(2), pp. 757-766.
70. Yu T., Lin K.-J. Service selection algorithms for Web services with end-to-end QoS constraints. Inform. Syst. and E-Business Manag., 2005, 3(2), pp. 103-126.
71. Zemel E. An O(n) algorithm for the linear multiple choice knapsack problem and related problems. Inform. Proc. Letters, 1984, 18(3), pp. 123-128.
72. Zitler E., Thiele L. Multiobjective evolutionary algorithms: A comparative study and the strength of Pareto approach. IEEE Trans. on Evol. Comput., 1999, 3(4), pp. 257-271.

Левин Марк Шмуилович. Старший научный сотрудник Института проблем передачи информации РАН, окончил Московский электротехнический институт связи в 1970 году и мех.-мат. факультет МГУ в 1975, кандидат технических наук, автор более 100 научных работ и трех монографий. Область научных интересов: системный анализ, проектирование систем, комбинаторная оптимизация, многокритериальное принятие решений, инженерное образование.

Сафонов Александр Валерьевич. Исследователь компании ТрансТелеКом, окончил Московский физико-технический институт (государственный университет) в 2008 году, автор 4 научных работ. Область научных интересов: системный анализ, проектирование систем, коммуникационные системы, комбинаторная оптимизация, многокритериальное принятие решений.