Задачи о рюкзаке

Модель размещения капитала

Дано $P_1, P_2, \ldots, P_N$ — проекты;

T — горизонт планирования (длина наиболее продолжительного проекта);

s_{tk} — доход от проекта P_k к концу года t;

y_{tk} — инвестиции в проект P_k в начале года t; $s_{0k} = y_{T+1,k} = 0$;

r — коэффициент дисконтирования затрат;

$b_k = \sum_{t=0}^{T} (s_{tk} - y_{t+1,k})/(1+r)^t$ — суммарная прибыль от проекта P_k;

$C = (c_1, \ldots, c_T)$ — доступный капитал для развития проектов

$A_k = (a_{1k}, \ldots, a_{Tk})$ — вектор затрат на реализацию проекта P_k (целые);

Если доход нельзя реинвестировать, то $a_{tk} = y_{tk}$, иначе $a_{tk} = y_{kt} - s_{t-1k}$.

Найти подмножество проектов, которые можно реализовать на капитал C и которые в сумме дают максимальную прибыль, то есть

$$\max \sum_{k=1}^{N} b_k x_k$$

при ограничениях:

$$\sum_{k=1}^{N} a_{tk} x_k \le c_t, \quad t = 1, \dots, T;$$

$$x_k \in \{0, 1\}, \quad k = 1, \dots, N.$$

Замечание 1. При $T = 1$ получаем линейную распределительную задачу с 0-1 переменными — задачу о рюкзаке.

Замечание 2. Без ограничения общности можно считать, что $\sum_{k=1}^{N} a_{tk} x_k \ge c_t$, $t = 1, \dots, T$; (можно получить задачу о рюкзаке даже при $T > 1$).

Алгоритм динамического программирования

Обозначим через $f_k(Y)$ максимальную прибыль от первых k проектов при доступном капитале $Y = (y_1, \dots, y_T)$.

Тогда

$$f_0(Y) = 0$$
$$f_{k+1}(Y) = \max[f_k(Y), b_{k+1} + f_k(Y - A_{k+1})],\ \ k = 0, \dots, N-1,\ \ 0 \le Y \le C,$$

где $f_k(Y - A_{k+1}) = -\infty$, если вектор $Y - A_{k+1}$ имеет хотя бы одну отрицательную компоненту.

$$\text{ТДП} = O(N \cdot c_1 \cdot \ldots \cdot c_T);$$

$$\text{ПДП} = O(N \cdot c_1 \cdot \ldots \cdot c_T).$$

Полный перебор — 2^N вариантов.

Верхняя оценка

Релаксация линейного программирования

$$\max \sum_{k=1}^{N} b_k x_k \tag{1}$$

при ограничениях

$$\sum_{k=1}^{N} a_{tk} x_k \le c_t, \quad t = 1, \dots, T; \tag{2}$$

$$0 \le x_k \le 1, \quad k = 1, \dots, N. \tag{3}$$

Теорема 2.1. Существует оптимальное решение x^{LP} с не более чем $\min(T, N)$ дробными компонентами

Доказательство. Пусть $T < N$ (иначе утверждение очевидно). Приведем задачу к канонической форме. Получим $2N + T$ переменных и $N + T$ ограничений:

$$\min \sum_{k=1}^{N} -b_k x_k \qquad (4)$$

$$\sum_{k=1}^{N} a_{tk} x_k + \lambda_t = c_t, \;\; t = 1, \ldots T; \qquad (5)$$

$$x_j + \mu_j = 1, \;\; j = 1, \ldots, N, \qquad (6)$$

$$x_j \geq 0, \;\; \mu_j \geq 0, \;\; \lambda_t \geq 0. \qquad (7)$$

Любое базисное допустимое решение имеет не менее N нулей. Предположим, что T из них соответствуют переменным λ_t. Тогда $N - T$ нулей останется для x_j и μ_j. Если для некоторого j имеем $\mu_j = 0$, то $x_j = 1$ — целое. Если $x_j = 0$ — тоже целое. Таким образом, получаем $N - T$ целых компонент для x_j, то есть T дробных. ■

Округление дробного решения

Пусть x^{LP} — оптимальное решение задачи (4)–(7). Для $\gamma \in [0,1)$ положим

$$x_j = \begin{cases} 1, \text{если } x_j^{LP} = 1, \\ 0, \text{если } x_j^{LP} < \gamma. \end{cases}$$

Для оставшихся дробных значений переменных сформируем подзадачу вида (4)–(7), пересчитав правые части ограничений. Найдем оптимальное решение x^{LP} для этой подзадачи и положим

$$x_j = \begin{cases} 1, & \text{если } x_j^{LP} = 1, \\ 0, & \text{если } x_j^{LP} = 0, \\ 0 & \text{для } j = \arg\min\{x_j^{LP} \mid 0 < x_j^{LP} < 1\}. \end{cases}$$

На этом шаге значение как минимум одной переменной будет зафиксировано. Повторяя процедуру, найдем допустимое решение исходной задачи.

Вопросы

- Пусть T— трудоемкость решения задачи линейного программирования. Оцените трудоемкость алгоритма округления дробного решения.

- Пусть Π — объем памяти для решения задачи линейного программирования. Оцените объем памяти алгоритма округления дробного решения.

- Пусть x^{LP} — оптимальное решение задачи линейного программирования. Если все компоненты вектора x^{LP} целые, то x^{LP} — оптимальное решение исходной целочисленной задачи (*Да или Нет?*)

- Если некоторые компоненты вектора x^{LP} целые, то эти целые значения сохранятся и в оптимальном решении исходной целочисленной задачи (*Да или Нет?*)

Задача об отправке грузов

$I = \{1, \dots, n\}$ — авиалайнеры, $J = \{1, \dots, m\}$ — контейнеры,

p_{ij} — доход от доставки авиалайнером i контейнера j,

w_j — вес контейнера j,

c_i — вместимость авиалайнера i,

$$x_{ij} = \begin{cases} 1, \text{если отправить контейнер } j \text{ авиалайнером } i \\ 0 \quad \text{иначе} \end{cases}$$

Модель

$$\max \sum_{i \in I} \sum_{j \in J} p_{ij} x_{ij}$$

при ограничениях:

$$\sum_{i \in I} x_{ij} \le 1, \quad j \in J;$$

$$\sum_{j \in J} w_j x_{ij} \le c_i, \quad i \in I;$$

$$x_{ij} \in \{0,1\}, i \in I, j \in J.$$

Дальняя экспедиция

Морское судно грузоподъемностью C отправляется в экспедицию.

$J = \{1, \dots, m\}$ — типы грузов (трактора, электрогенераторы, радиостанции,…)

N_j — варианты грузов для $j \in J$, w_{ij} — вес груза j по варианту $i \in N_j$

p_{ij} — полезность груза

$$x_{ij} = \begin{cases} 1, & \text{если берем } i\text{-й вариант груза } j \\ 0 & \text{иначе} \end{cases}$$

Модель

$$\max \sum_{j \in J} \sum_{i \in I} p_{ij} x_{ij}$$

при ограничениях:
$$\textstyle\sum_{i \in N_j} x_{ij} = 1, \dots j \in J;$$

$$\textstyle\sum_{j \in J} \sum_{i \in N_j} w_{ij} x_{ij} \le C;$$

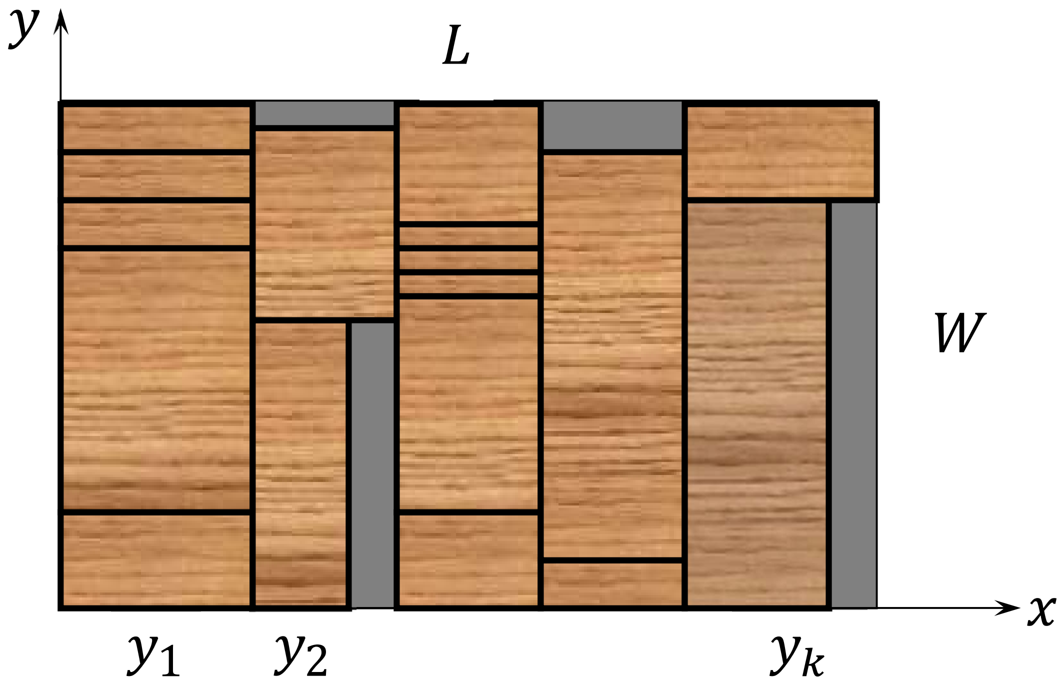
$$x_{ij} \in \{0,1\}, \;\; i \in N_j, \;\; j \in J.$$

Гильотинный раскрой материала

Дан лист размера $L \times W$ и n типов прямоугольников $l_j \times w_j$, $j = 1, \ldots, n$

$p_j > 0$ — доход от прямоугольника j, повороты запрещены, разрезы параллельно осям координат от кромки до кромки. Двухстадийная обработка: сначала режем лист параллельно оси y, затем параллельно оси x.

Найти раскрой листа с максимальным доходом.



Пусть

k — число параллельных полос $k = \lfloor L/l_{\min} \rfloor$

y_i — ширина полосы i, $1 \le i \le k$,

x_{ij} — число j-х прямоугольников в полосе i,

$$x'_{ij} = \begin{cases} 1, & \text{если } x_{ij} > 0 \\ 0 & \text{иначе} \end{cases}$$

$m_j = \lfloor W / w_j \rfloor$ — максимально возможное число j-х прямоугольников в полосе.

Модель:

$$\max \sum_{i=1}^{k} \sum_{j=1}^{n} p_j x_{ij}$$

при ограничениях

$$\sum_{j=1}^{n} w_j x_{ij} \le W, \quad i = 1, \dots, k;$$

$$\sum_{i=1}^{k} y_i \le L;$$

$$l_j x'_{ij} \le y_i, \quad i = 1, \dots, k, \quad j = 1, \dots, n;$$

$$m_j x'_{ij} \ge x_{ij}, \quad i = 1, \dots, k, \quad j = 1, \dots, n;$$

$$x'_{ij} \in \{0,1\}, \quad x_{ij} \in \{0, \dots, m_j\}, \quad y_i \ge 0.$$

Классическая задача о рюкзаке

Найти:

$$\max \sum_{j \in J} p_j x_j$$

при ограничениях

$$\sum_{j \in J} w_j x_j \leq C;$$

Все коэффициенты p_j, w_j, C — целые числа. $x_j \in \{0,1\},\ j \in J$.

Определение Алгоритм *A* называется *приближенным алгоритмом с гарантированной абсолютной точностью* K, если для любого примера I алгоритм находит значение $z^A(I)$ с отклонением от оптимума $z^*(I)$ не более K, то есть

$$z^*(I) - z^A(I) \leq K, \quad \text{для всех } I.$$

Обозначим через $T_A(n, C)$ трудоемкость алгоритма A для задачи с n предметами и вместимостью рюкзака C.

Теорема 2.2. Пусть A — приближенный алгоритм с гарантированной абсолютной точностью K и трудоемкостью $T_A(n, C)$. Тогда алгоритм A для любого примера позволяет найти точное решение задачи о рюкзаке с той же трудоемкостью.

Доказательство. Пример I задается числами $p_1, \dots, p_n$, $w_1, \dots, w_n$, C. Построим новый пример I' положив $C' = C$, $p_j' = (K+1)p_j$, $w_j' = w_j$, $j \in J$. Оба примера имею одно и то же множество допустимых решений. Так как целевая функция для I' в $(K+1)$ раз больше, чем для I, то оптимальные наборы x_j^* совпадают.

Для примера I' имеем

$z^*(I') - z^A(I') \le K$, но $z^A(I') = (K+1)z^A(I)$ и $z^*(I') = (K+1)z^*(I)$, то есть

$$z^*(I) - z^A(I) \le \frac{K}{K+1}.$$

Так как p_j — целые, то $z^*(I) - z^A(I) \le 0$, то есть $z^*(I) = z^A(I)$, что и требовалось доказать. ■

Жадные алгоритмы

Упорядочим предметы по плотности $\frac{p_j}{w_j}$ и будем считать, что

$$\frac{p_1}{w_1} \geq \frac{p_2}{w_2} \geq \cdots \geq \frac{p_n}{w_n}.$$

Жадный алгоритм

1. $\bar{w} := 0;\ \ z^G := 0;$
2. $\text{for } j := 1 \text{ to } n \text{ do}$

   $\text{if } \ \bar{w} + w_j \leq C \ \text{ then}$

   $x_j := 1;\ \bar{w} := \bar{w} + w_j;\ z^G := z^G + p_j;$

   $\text{else } x_j := 0;$

$$T_G = O(n \log n + n), \quad \Pi_G = O(n)$$

Упражнение. Если последнюю строку заменить на

$$\text{else} \{ \text{for } k := j \text{ to } n \text{ do } x_k := 0;\ break \},$$

то такое решение можно найти с $T = O(n)$.

Релаксация линейного программирования

LP–релаксация

$$z^{LP} = \max \sum\nolimits_{j \in J} p_j x_j$$

$$\sum_{j \in J} w_j x_j \le C;$$

$$0 \le x_j \le 1, \ \ j \in J.$$

Так как область допустимых решений увеличилась, то $z^{LP} \ge z^*$. Пусть предметы упорядочены по плотностям и для некоторого $s \in J$ верно:

$$\sum_{j=1}^{s-1} w_j \le C \quad \text{и} \quad \sum_{j=1}^{s} w_j > C.$$

Положим

$$x_j^{LP} = \begin{cases} 1, & j = 1, \dots, s-1, \\ \dfrac{1}{w_s}\left(C - \sum_{j=1}^{s-1} w_j\right) \\ 0, & j = s+1, \dots, |J|. \end{cases}$$

Теорема 2.3. Решение x^{LP} является оптимальным решением LP–релаксации и

$$z^{LP} = \sum_{j=1}^{s-1} p_j + \frac{p_s}{w_s}\left(C - \sum_{j=1}^{s-1} w_j\right).$$

Доказательство. Будем считать, что предметы с одинаковой плотностью слиты в один и

$$\frac{p_1}{w_1} > \frac{p_2}{w_2} > \cdots > \frac{p_n}{w_n}.$$

Пусть $\bar{x} = (\bar{x}_1, \dots, \bar{x}_n)$ — оптимальное решение, не равное x^{LP}. Так как $\sum_{j\in J} w_j \bar{x}_j = C$, то найдутся как минимум два номера $k > s$ и $i \le s$ такие, что $\bar{x}_k > 0$ и $\bar{x}_i < x_i^{LP}$. Положим $d = \min\{w_k \bar{x}_k,\ w_i(x_i^{LP} - \bar{x}_i)\} > 0$. Построим новое решение x', которое будет отличаться от $\bar{x}$ только в координатах i, k:

$$x_i' = \bar{x}_i + \frac{d}{w_i}, \qquad x_k' = \bar{x}_k - \frac{d}{w_k}.$$

Решение x' является допустимым, так как $x_j' \geq 0$ по выбору d и

$$\sum_{j \in J} w_j x_j' = \sum_{j \in J} w_j \bar{x}_j + \frac{w_i d}{w_i} - \frac{w_k d}{w_k}$$

Кроме того,

$$\sum_{j \in J} p_j x_j{}' = \sum_{j \in J} p_j \bar{x}_j + d\left(\frac{p_i}{w_i} - \frac{p_k}{w_k}\right) > \sum_{j \in J} p_j \bar{x}_j$$

так как $\frac{p_i}{w_i} > \frac{p_k}{w_k}$, что противоречит оптимальности $\bar{x}$. ■

Свойства *LP*-релаксации

Верхняя оценка $U^{LP} = \lfloor z^{LP} \rfloor$, $\qquad \hat{p} = \sum_{j=1}^{s-1} p_j$

Свойство 1. $\quad \hat{p} \le z^* \le U^{LP} \le z^{LP} \le \sum_{j=1}^{s} p_j \le \hat{p} + p_s \le z^G + p_s.$

Свойство 2. $\quad z^* - z^G \le z^* - \hat{p} \le p_{max}$, где $p_{max} = \max_{j \in J} p_j.$

Свойство 3. $\quad z^{LP} \le 2z^*$ и для любого $\varepsilon > 0$ найдется пример задачи о рюкзаке такой, что $z^{LP} \ge 2z^* - \varepsilon$.

Доказательство. 1. Так как $z^* \ge \sum_{j=1}^{s-1} p_j$ и $z^* \ge p_s$ то $2z^* \ge z^{LP}$.

2. Рассмотрим пример $n = 2, C = 2M$ и $w_j = M + 1, p_j = 1, j = 1,2.$

Тогда $z^* = 1$, но $z^{LP} = \frac{2M}{M+1}$ и с ростом M получаем $z^{LP} / z^* \to 2$. ■

Определение. Алгоритм A называется *приближенным алгоритмом с гарантированной относительной точностью* K, если для любого примера I алгоритм находит значение $z^A(I)$ такое, что $\frac{z^A(I)}{z^*(I)} \geq K$ для всех I.

Если $\varepsilon = 1 - K$, то $\frac{z^*(I) - z^A(I)}{z^*(I)} \leq \varepsilon$ – относительная погрешность алгоритма.

Пример. Положим $n = 2, C = M$ и $p_1 = 2$, $w_1 = 1$, $p_2 = M$, $w_2 = M$. Тогда жадный алгоритм получит $x_1 = 1$, $x_2 = 0$, $z^A = 2$,

но $x_1^* = 0$, $x_2^* = 1$, $z^* = M$, то есть для жадного алгоритма

$$\frac{z^A}{z^*} \to 0 \text{ при } M \to \infty.$$

Модифицированный жадный алгоритм

Используем предыдущий жадный алгоритм, получаем z^G.

Затем полагаем $z^{MG} = \max\{z^G, \max\{p_j \,|\, j \in J\}\}$.

Теорема 2.4. Модифицированный жадный алгоритм A^{MG} имеет гарантированную относительную точность $K = ½$.

Доказательство. Из свойства 2 для LP-релаксации имеем

$$z^* \le z^G + p_{max} \le z^{MG} + z^{MG}. \quad ■$$

Пример. Положим $n = 3,\ C = 2M$,

$$p_1 = 2,\ p_2 = M,\ p_3 = M,$$
$$w_1 = 1,\ w_2 = M,\ w_2 = M.$$

Получаем $z^* = 2M$, $z^{MG} = 2 + M$, то есть оценку $K = ½$ нельзя улучшить.

Алгоритм $G^{3/4}$

Сокращаем погрешность за счет трудоемкости

Алгоритм $G^{3/4}$

1. $z^A := \max\{p_j \mid j \in J\}$;
2. Для всех пар $(i, k) \in J \times J$

   if $w_i + w_k \le C$ then

   - применяем алгоритм A^{MG} к задаче с множеством $\{ j \mid p_j \le \min\{p_i, p_k\} \} \setminus \{i, k\}$ и вместимостью рюкзака $C - w_i - w_k$
   - if $p_i + p_k + z^{MG} > z^A$ then $z^A := p_i + p_k + z^{MG}$.

Теорема 2.5. Алгоритм $G^{3/4}$ имеет гарантированную относительную точность $K = 3/4$.

Доказательство. Если оптимальное решение x_j^* содержит только один предмет, то $z^A = z^*$ и утверждение верно. Предположим, что в оптимальном решении не меньше двух предметов. Выберем среди них два (i_*, k_*) с наибольшими p_j. На некотором шаге алгоритм $G^{3/4}$ выберет эту пару (i_*, k_*) и применит алгоритм A^{MG} к задаче с множеством предметов $\{j \mid p_j \le \min\{p_{i_*}, p_{k_*}\}\}\backslash\{i_*, k_*\}$ и вместимостью рюкзака $C - w_{i_*} - w_{k_*}$.

Обозначим через z_S^* оптимальное решение этой подзадачи. Тогда $z^* = p_{i_*} + p_{k_*} + z_S^*$. Алгоритм A^{MG} для этой подзадачи найдет значение z_S^{MG}. Так как z^A — лучшее из решений, просмотренных алгоритмом $G^{3/4}$, то $z^A \ge p_{i_*} + p_{k_*} + z_S^{MG}$

По теореме 2.4 имеем $z_S^{MG} \ge \frac{1}{2} z_S^*$.

Рассмотрим два случая.

Случай 1. $p_{i_*} + p_{k_*} \geq \frac{1}{2} z^*$. Тогда $z^A \geq p_{i_*} + p_{k_*} + z_S^{MG} \geq p_{i_*} + p_{k_*} + \frac{1}{2} z_S^* =$ $p_{i_*} + p_{k_*} + \frac{1}{2}\left(z^* - p_{i_*} - p_{k_*}\right) = \frac{1}{2}\left(z^* + p_{i_*} + p_{k_*}\right) \geq \frac{3}{4} z^*$.

Случай 2. $p_{i_*} + p_{k_*} < \frac{1}{2} z^*$. Тогда $\min\left(p_{i_*}, p_{k_*}\right) < \frac{1}{4} z^*$. По определению z_S^* содержит предметы с $p_j \leq \frac{1}{4} z^*$, значит $z_S^* \leq z_S^{LP} \leq z_S^{MG} + \frac{1}{4} z^*$.

Теперь $z^* = p_{i_*} + p_{k_*} + z_S^* \leq p_{i_*} + p_{k_*} + z_S^{MG} + \frac{1}{4} z^* \leq z^A + \frac{1}{4} z^*$. ■

Пример. Положим $n = 5,\ C = 4M$,

$$p_1 = 2,\ \ p_2 = p_3 = p_4 = p_5 = M,$$
$$w_1 = 1,\ w_2 = w_3 = w_4 = w_5 = M.$$

Очевидно, что $z^* = 4M,\ z^A = 3M + 2,$ то есть оценку $K =$ ¾ нельзя улучшить.

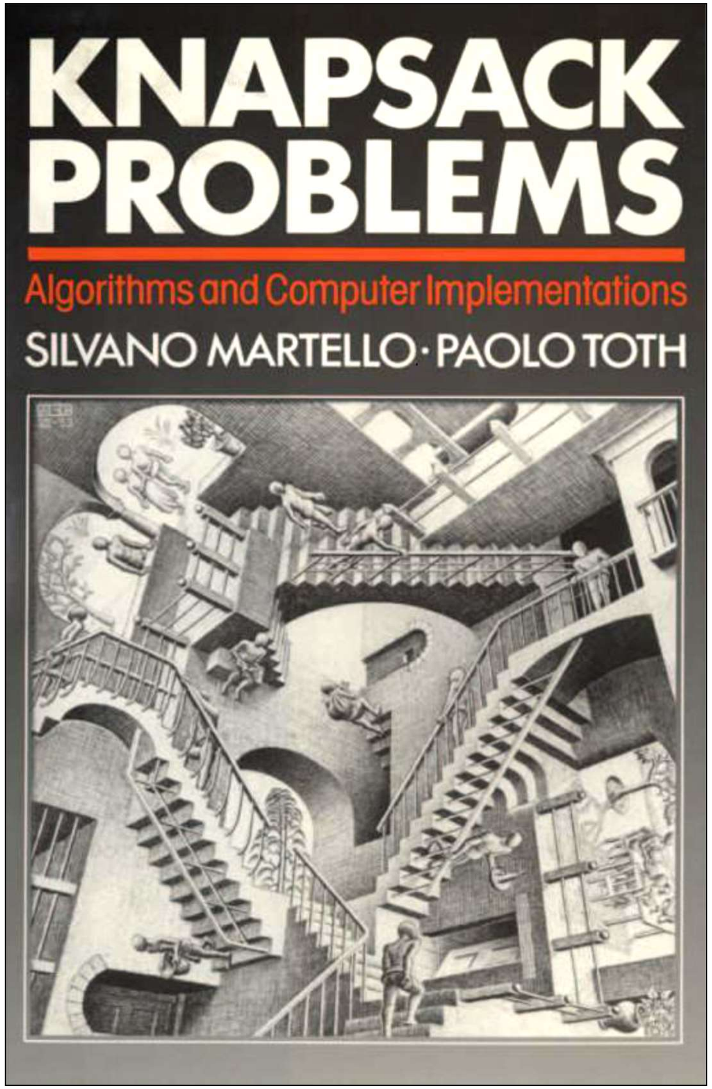
Silvano Martello, Paolo Toth

Knapsack Problem

Algorithms and Computer Implementations

University of Bologna

John Wiley & Sons. 1990 . 296 p



http://www.math.nsc.ru/LBRT/k5/knapsack_problem.pdf

Вопросы

- Алгоритм $G^{3/4}$ для задачи о рюкзаке является полиномиальным *(Да или Нет?)*

- Алгоритм $G^{3/4}$ требует $O(n)$ затрат памяти *(Да или Нет?)*

- Алгоритм $G^{3/4}$ становится точным, если вместо алгоритма A^{MG} использовать на шаге 2.1 точный алгоритм для оставшегося места в рюкзаке и подмножества предметов $\{j \mid p_j \le \min\{p_i, p_k\}\}\backslash\{i, k\}$ *(Да или Нет?)*

- Алгоритм $G^{3/4}$ становится точным, если вместо алгоритма A^{MG} использовать на шаге 2.1 точный алгоритм для оставшегося места в рюкзаке и подмножества предметов без пары $\{i, k\}$ *(Да или Нет?)*

- Можно ли за полиномиальное время получить точность 0.9 *(Да или Нет?)*