Свиридова О.В., Рыбанов А.А.

Целочисленное программирование

Волжский

2022

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ВОЛЖСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ (ФИЛИАЛ)
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО ОБРАЗОВАТЕЛЬНОГО
УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ
«ВОЛГОГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
КАФЕДРА «ИНФОРМАТИКИ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ»

О.В. Свиридова, А.А. Рыбанов

Целочисленное программирование

Электронное учебное пособие

Волжский

2022

УДК 004.42(07)
ББК 32.973я73
С 247

Рецензенты:

заведующий кафедрой математики, информатики и естественных наук Волжского филиала Волгоградского государственного университета, канд. физ.-мат. наук, доцент
Полковников А.А.

канд. педагогических наук, доцент
кафедры методики преподавания математики и физики, ИКТ
ФГБОУ ВО «Волгоградский государственный социально-педагогический университет (ВГСПУ)»
Филиппова Е.М.

Издается по решению редакционно-издательского совета
Волгоградского государственного технического университета

Свиридова, О.В.

Целочисленное программирование [Электронный ресурс] : учебное пособие / О. В. Свиридова, А. А. Рыбанов ; Министерство науки и высшего образования Российской Федерации, ВПИ (филиал) ФГБОУ ВО ВолгГТУ. – Электронные текстовые данные (1 файл: 184 КБ). – Волжский, 2022. – Режим доступа: http://lib.volpi.ru. – Загл. с титул. экрана.

ISBN 978-5-9948-4411-3

Пособие имеет учебно-методическое назначение и предназначено для изучения студентами раздела «Целочисленное программирование» дисциплины учебного плана «Методы оптимизации». В пособии рассмотрены методы решения и примеры целочисленных задач.

Предназначено для студентов, обучающихся по направлению 09.03.01 «Информатика и вычислительная техника» в рамках курса «Методы оптимизации».

Илл. 6, табл. 21, библиограф.: 6 назв.

ISBN 978-5-9948-4411-3

Содержание

ВВЕДЕНИЕ

Электронный ресурс «Целочисленное программирование» является учебным пособием, которое содержит теоретический материал по линейному целочисленному программированию и является дополнением к лекциям по дисциплине «Методы оптимизации».

Целочисленное программирование ориентировано на решение задач математического программирования, в которых все или некоторые переменные должны принимать только целочисленные значения. Задача называется полностью целочисленной, если условие целочисленности наложено на все ее переменные; когда это условие относится лишь к некоторым переменным, задача называется частично целочисленной. Если при этом целевая функция и функции, входящие в ограничения, линейные, то говорят, что данная задача является задачей целочисленного линейного программирования (ЦЛП).

Целочисленные задачи математического программирования возникают различными путями:

1. Существуют задачи линейного программирования, которые формально к целочисленным не относятся, но при соответствующих исходных данных всегда обладают целочисленным планом. Примеры таких задач – транспортная задача и ее модификации (задачи о назначениях, о потоках в сетях).

2. Толчком к изучению целочисленных задач в собственном смысле слова явилось рассмотрение задач линейного программирования, в которых переменные представляли физически неделимые величины. Они были названы задачами с неделимостью, например, задачи об оптимизации комплекса средств доставки грузов, о нахождении минимального порожнего пробега автомобилей при выполнении заданного плана перевозок, об определении оптимального машинного парка и его оптимального распределения по указанным работам при условии минимизации суммарной стоимости (машинного парка и производимых работ) и т.п.

3. Другим важным толчком к построению теории целочисленного программирования стал новый подход к некоторым экстремальным комбинаторным задачам. В них требуется найти экстремум целочисленной линейной функции, заданной на конечном множестве элементов. Такие задачи принято называть задачами с альтернативными переменными. В качестве примеров можно назвать задачи коммивояжера, об оптимальном назначении, теории расписания и задачи с дополнительными логическими условиями (например, типа «или – или», «если – то» и т.п.).

Исторически первой задачей целочисленного типа является опубликованная в 1932 г. венгерским математиком Е. Эгервари задача о назначении персонала. В 1955 г. на Втором симпозиуме по линейному программированию была рассмотрена задача о бомбардировщике, известная как задача о ранце.

В большинстве случаев целочисленные задачи сильно отличаются от своих непрерывных аналогов и требуют для решения специальных методов.

Условно методы решения задач целочисленного программирования можно разделить на три основных группы: методы отсечения, комбинаторные методы, приближенные методы. При наличии в задаче целочисленного линейного программирования двух переменных в системе ограничений неравенств задача может быть решена графическим методом.

Данное электронное учебное пособие разработано в соответствии с требованиями Федерального государственного стандарта высшего профессионального образования и предназначены для студентов бакалавриата, обучающихся по направлению 09.03.01 «Информатика и вычислительная техника». Областью применения данного электронного ресурса является учебный процесс в Волжском политехническом институте (филиал) Волгоградского технического университета, объектом обучения могут выступать студенты направления «Информатика и вычислительная техника», изучающие дисциплину «Методы оптимизации», а также студенты направления «Программная инженерия», изучающие дисциплину «Исследование операций».

1. Методы целочисленного линейного программирования

1.1. Постановка задачи целочисленного программирования

Целочисленное линейное программирование (ЦЛП) ориентировано на решение задач линейного программирования (ЛП), в которых на все или часть переменных наложены условия целочисленности. Различают полностью целочисленные задачи и частично целочисленные задачи.

К необходимости рассматривать целочисленные задачи мы приходим по двум причинам:

1) существует много практических задач, которые изначально по своей природе являются целочисленными;

2) существуют задачи, которые явно не содержат требований целочисленности, но содержат условия, которые трудно или невозможно (в исходной постановке) записать в виде формул (т.е. математически формализовать). Это так называемые «некорректные» задачи. Анализ таких задач, введение новых целочисленных переменных позволяют преодолеть эту трудность и свести задачу к стандартному виду.

В общей постановке задача ЦЛП имеет вид

$$c'x \to max, \tag{1}$$

$$Ax = b,\ d_{*j} \le x_j \le d_j^*,\ j \in J, \tag{2}$$

$$x_j - \text{целое},\ j \in I, \tag{3}$$

где $x = (x_j, j \in J)$, $J = \{1,2,\ldots,n\}$, $I \subset J$; $A \in R^{m \times n}$, rank $A = m$, $b \in R^m$, $c \in R^n$.

Пример. Рассмотрим задачу

$$21x_1 + 11x_2 \to \max, \tag{4}$$

$$7x_1 + 4x_2 \le 13,\ x_1 \ge 0, x_2 \ge 0, \tag{5}$$

$$x_1, x_2 - \text{целые}. \tag{6}$$

Эта задача имеет единственный оптимальный план x^0 =(x_1 = 0, x_2 = 3) (рис. 1.1).

Существует такое мнение, что задачи ЦЛП можно решать следующим образом: надо решить задачу без предположения целочисленности, а потом «округлить» полученное нецелочисленное решение до целочисленного.

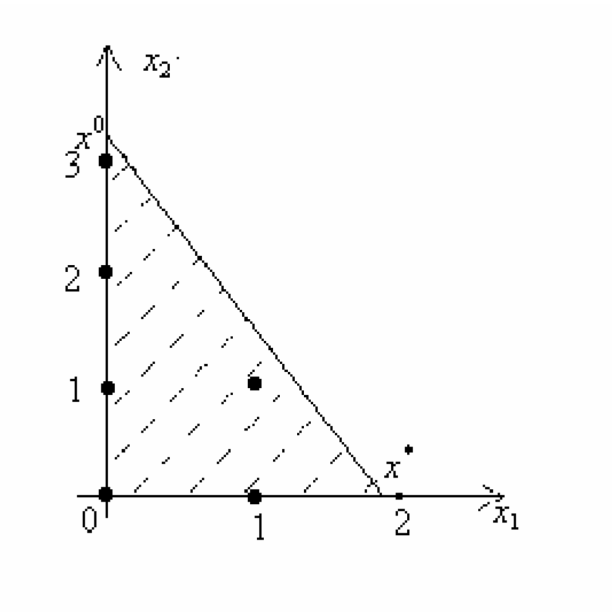


Рис. 1.1

Посмотрим, что получится, если воспользуемся этим приемом. Оптимальное решение нецелочисленной задачи (4), (5) имеет вид x^*=(x_1= $\frac{13}{7}$, x_2 = 0). Очевидное округление приводит к вектору (x_1= 2, x_2= 0), который не является даже планом.

Округление в «меньшую сторону» приводит к плану (x_1 = 1, x_2 = 0). Этот план допустимый, но очень далек от оптимального.

Таким образом, мы видим, что «метод округления» нельзя считать универсальным.

Еще более неприятная особенность, свойственная задачам целочисленного программирования, состоит в том, что нет простого способа, позволяющего определить, является ли данный допустимый план оптимальным. В этом одно из главных отличий задач ЦЛП от задач ЛП. Чтобы проиллюстрировать это, предположим, что в задаче (4) – (6) надо проверить, является ли план ($x_1 = 1$, $x_2 = 1$) оптимальным. По аналогии с ЛП проверим, соответствует ли точка ($x_1 = 1$, $x_2 = 1$) какому-либо локальному оптимуму в том смысле, что значение целевой функции не улучшится при переходе в любую соседнюю допустимую целочисленную точку:

$$x_1 = 1 + d,\ x_2 = 1 + l, \text{ где } d,\ l = -1 \vee 0 \vee 1.$$

В нашем примере допустимые соседние точки имеют вид

$$(x_1 = 0, x_2 = 0), (x_1 = 0, x_2 = 1), (x_1 = 0, x_2 = 2), (x_1 = 1, x_2 = 0). \qquad (7)$$

Легко проверить, что значение критерия качества в точке ($x_1 = 1$, $x_2 = 1$) лучше, чем в точках (7). Однако вектор ($x_1 = 1$, $x_2 = 1$) не является оптимальным планом.

Отметим, что для задач с булевыми переменными $x_j = 0 \vee 1$ проверка всех соседних дискретных точек сводится к полному перебору всех возможных планов. Возникает вопрос: существуют ли методы решения задач ЦЛП, отличные от полного перебора, либо полный перебор – это единственный способ решения?

Если задача имеет небольшую размерность, то метод полного перебора можно использовать. Однако, если размеры большие, то полный перебор по времени выливается приблизительно «в целую жизнь». Например, при рассмотрении задачи ЦЛП, в которой 100 переменных и $x_j = 0 \vee 1$, для полного перебора надо перебрать 2^{100} вариантов. Следовательно, нужны другие методы, отличные от полного перебора.

К настоящему времени наиболее популярными методами целочисленного программирования являются методы двух типов:

1) возврата;
2) отсекающих плоскостей.

Рассматриваемые ниже, в частности, метод ветвей и границ относится к первой группе, а метод Гомори относится ко второй группе методов.

1.2. Графический метод

Если в задаче целочисленного линейного программирования целевая функция и система ограничений-неравенств зависят от двух переменных, то задача может быть решена **графическим методом**.

В системе координат $X_1 0 X_2$ находят область допустимых решений, строят вектор C и линию уровня. Перемещая линию уровня по направлению C для задач на максимум, определяют наиболее удаленную от начала координат точку и ее координаты.

В случае, когда координаты этой точки нецелочисленные, в области допустимых решений строят целочисленную решетку и находят на ней такие целые числа x_1 и x_2, которые удовлетворяют системе ограничений и при которых значение целевой функции наиболее близко к экстремальному нецелочисленному решению. Координаты такой вершины и являются целочисленным решением.

Аналогично решается задача на минимум. Целочисленному минимуму целевой функции будет соответствовать координата вершины целочисленной решетки, лежащей в области допустимых решений, наиболее близкой к началу координат в направлении вектора C.

Рассмотрим алгоритм решения задачи целочисленного программирования на конкретном примере.

Задача 1.1. Для улучшения финансового положения фирма приняла решение об увеличении выпуска конкурентоспособной продукции, для чего в одном из цехов необходимо установить дополнительное оборудование, требующее $\frac{19}{3}$ м^2 площади. На приобретение дополнительного оборудования фирма выделила 10 тыс. ден. ед., при этом она может купить оборудование двух видов. Приобретение одного комплекта оборудования 1-го вида стоит 1 тыс. ден. ед., 2-го вида – 3 тыс.

ден. ед. Приобретение одного комплекта оборудования 1-го вида позволяет увеличить выпуск продукции в смену на 2 ед., а одного комплекта оборудования 2-го вида – на 4 ед. Зная, что для установки одного комплекта оборудования 1-го вида требуется 2 м^2 площади, а для оборудования 2-го вида – 1 м^2 площади, определить такой набор дополнительного оборудования, который дает возможность максимально увеличить выпуск продукции.

Решение. Предположим, что фирма приобретае x_1 комплектов дополнительного оборудования 1-го вида и x_2 комплектов оборудования 2- го вида.

Математическая модель задачи в этом случае имеет вид

$$F(x) = 2x_1 + 4x_2 \to \max$$

$$\begin{cases} 2x_1 + x_2 \le \dfrac{19}{3} \\ x_1 + 3x_2 \le 10 \end{cases},$$

$$x_1 \ge 0,\ x_2 \ge 0,\ x_1, x_2 \in Z.$$

Получим задачу целочисленного программирования. Так как неизвестных только два (x_1 и x_2), то найдем ее решение графическим способом (рис. 1.2).

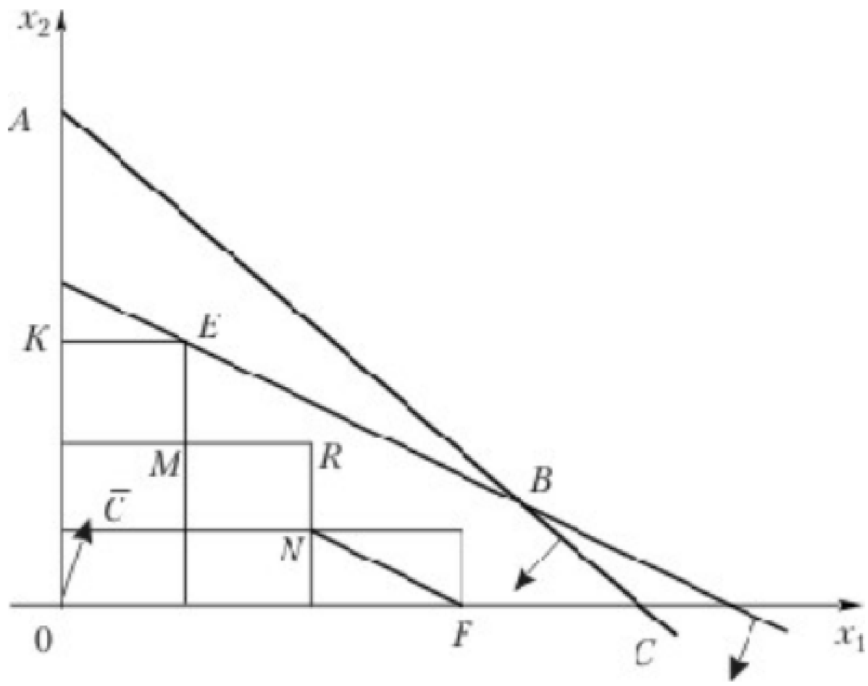


Рис. 1.2

OABC – область допустимых решений (ОДР). Оптимальное решение задача имеет в точке ***B*** (9/5, 41/15) при этом максимальное значение целевой функции составляет 218/15 ед. Полученное оптимальное решение не целочисленное.

Условию целочисленности переменных удовлетворяют координаты 12 точек, принадлежащих ОДР. Чтобы найти точку, координаты которой определяют решение исходной задачи, заменим многоугольник ***OABC*** многоугольником ***OKEMNF***, содержащим все допустимые точки с целочисленными координатами.

Строим вектор $\overline{C}$ (2; 4). Линию уровня перемещаем по направлению $\overline{C}$, получим в точке Е (1, 3) максимальное значение целевой функции $F = 2\cdot 1 + +4\cdot 3 = 14$ (ед.).

Ответ. Фирме следует приобрести один комплект оборудования 1-го вида и три комплекта оборудования 2-го вида, что обеспечит ей при имеющихся ограничениях на производственные площади и денежные средства максимальное увеличение выпуска продукции, равное 14 ед. в смену.

Итак, один из простейших подходов при решении задач целочисленного программирования заключается в решении непрерывной модификации целочисленной задачи с последующим округлением координат полученного оптимума до допустимых целых значений. Округление в данном случае есть не что иное, как приближение. Например, если получено решение непрерывной задачи, содержащее число 10,1, то можно приближенно заменить это число на 10 (т. е. округлить до 10). Однако нет гарантии, что округленное решение будет удовлетворять ограничениям. Например, для линейной задачи с ограничениями в виде равенств округленное решение всегда не удовлетворяет этим ограничениям. Как следует из теории линейного программирования, округленное решение не может быть допустимым, поскольку это означало бы, что один и тот же базис (при условии равенства нулю небазисных переменных) определяет два различных решения задачи.

Приведем пример, который показывает возможную несостоятельность подхода, основанного на округлении оптимального решения задачи линейного программирования, полученной из исходной целочисленной задачи путем отбрасывания требований целочисленности.

Задача 1.2. Решить следующую задачу ЦЛП:

$$F(x) = -5x_1 - 2x_2 \to \min$$
$$\begin{cases} 11x_1 + 4x_2 \le 33 \\ -x_1 + 2x_2 \le 8 \end{cases},$$
$$x_1 \ge 0,\ x_2 \ge 0,\ x_1, x_2 \in Z.$$

Решение. Отбросив требование целочисленности переменных, получим задачу линейного программирования. Решением данной задачи является точка $x^*=(x^*_1,\ x^*_2)=\left(1\frac{4}{13}, 4\frac{17}{26}\right)$. Если рассматривать все возможные округления то получим четыре возможных целочисленных точки (1, 4), (2, 4), (1, 5), (2, 5). Во-первых, из этих точек только точка (1, 4) является допустимой для исходной задачи. А во-вторых, она находится далеко от истинного оптимального решения исходной задачи $x^*=(x^*_1,\ x^*_2)=(3;\ 0)$.

Невозможно обосновать какое-либо утверждение относительно близости целочисленного решения и решения задачи без требований целочисленности, поскольку можно построить пример, показывающий, что данный подход может давать сколь угодно плохие приближения к решению исходной целочисленной задачи.

Эффект округления не слишком заметен, лишь когда искомые параметры задачи подчинены относительно нежестким ограничениям. Однако типичными для задач целочисленного программирования являются ограничения-равенства, достаточно жестко определяющие поведение релевантных переменных. Примером может служить условие $x_1+x_2+...+x_n=1$, где $x_i=(0,1)$ для всех *i*. В такой ситуации процедура округления не имеет смысла, и необходимо обратиться к другим алгоритмам решения.

Несостоятельность округления подчеркивается также следующими соображениями. Несмотря на то, что целочисленные переменные обычно выражают количество неделимых предметов (например, машин, людей, кораблей), возможны и другие типы спецификации этих переменных. Так, решение о финансировании некоторого проекта представляется булевой переменной x (x=0, если проект отклоняется, и x=1, если проект принимается). В этом случае бессмысленно оперировать дробными значениями величины x, и процедура округления является логически неприемлемой.

Огромное количество экономических задач носит дискретный, чаще всего целочисленный, характер, что связано, как правило, с физической неделимостью многих элементов расчета: например, нельзя построить два с половиной завода, купить полтора автомобиля и т.д. В ряде случаев такие задачи решаются обычными методами, например, симплексным методом, с последующим округлением до целых чисел. Однако такой подход оправдан, когда отдельная единица составляет очень малую часть всего объема (например, товарных запасов); в противном случае он может внести значительные искажения в действительно оптимальное решение. Поэтому разработаны специальные методы решения целочисленных задач, например, метод ветвей и границ.

1.3. Метод ветвей и границ

Пусть

$$F = \sum_{j=1}^{n} c_j x_j \to \max \tag{7}$$

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i, \ i = 1, \ldots, m, \tag{8}$$

$$x_j \ge 0, \ x_j \in Z, \ j = 1, \ldots, n. \tag{9}$$

Первоначально находим симплексным методом или методом искусственного базиса оптимальный план задачи без учета целочисленности переменных.

Если среди компонент этого плана нет дробных чисел, то тем самым найдено искомое решение данной задачи.

Если среди компонент плана имеются дробные числа, то необходимо осуществить переход к новым планам, пока не будет найдено решение задачи.

Метод ветвей и границ основан на предположении, что оптимальный нецелочисленный план дает значение функции большее, чем всякий последующий план перехода.

Пусть переменная x^* в плане – дробное число. Тогда в оптимальном плане ее значение будет, по крайней мере:

1. либо меньше или равно ближайшему меньшему целому числу K^*_i;
2. либо больше или равно ближайшему большему целому числу K^*_i+1.

Определяя эти числа, находим симплексным методом решение двух задач линейного программирования:

$$F=\sum_{j=1}^{n} c_j x_j \to \max \tag{10}$$

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i,\ i=1,\dots,m, \tag{11}$$

$$x_i^* \le K_i^* \tag{12}$$

$$x_j \ge 0,\ x_j \in Z,\ j=1,\dots,n. \tag{13}$$

и

$$F=\sum_{j=1}^{n} c_j x_j \to \max \tag{14}$$

$$\sum_{j=1}^{n} a_{ij} x_j \le b_i,\ i=1,\dots,m, \tag{15}$$

$$x_i^* \ge K_i^*+1 \tag{16}$$

$$x_j \ge 0,\ x_j \in Z,\ j=1,\dots,n. \tag{17}$$

Возможны четыре случая при решении этой пары задач:

1. Одна из задач неразрешима, а другая имеет целочисленный оптимальный план. Тогда этот план и значение целевой функции дают решение исходной задачи.

2. Одна из задач неразрешима, а другая имеет нецелочисленный оптимальный план. Тогда рассматриваем вторую задачу и в ее оптимальном плане выбираем

одну из компонент, значение которой равно дробному числу, и строим две задачи, аналогичные предыдущим.

3. Обе задачи разрешимы. Одна из задач имеет оптимальный целочисленный план, а в оптимальном плане другой задачи есть дробные числа. Тогда вычисляем значения целевой функции от планов и сравниваем их между собой. Если на целочисленном оптимальном плане значение целевой функции больше или равно ее значению на плане, среди компонент которого есть дробные числа, то данный целочисленный план является оптимальным для исходной задачи и дает искомое решение.

4. Обе задачи разрешимы, и среди оптимальных планов обеих задач есть дробные числа. Тогда рассматриваем ту из задач, для которой значение целевой функции является наибольшим. И строим две задачи.

Алгоритм метода ветвей и границ:

1. Находим решение задачи линейного программирования без учета целочисленности.

2. Составляем дополнительные ограничения на дробную компоненту плана.

3. Находим решение двух задач с ограничениями на компоненту.

4. Строим в случае необходимости дополнительные ограничения, согласно возможным четырем случаям получаем оптимальный целочисленный план либо устанавливаем неразрешимость задачи.

Задача 1.3. Решить следующую задачу ЦЛП:

$$F = 2x_1 + x_2 \to \max$$

$$\begin{cases} 6x_1 + 4x_2 + x_3 = 24 \\ 3x_1 - 3x_2 + x_4 = 9 \\ -x_1 + 3x_3 + x_5 = 3 \end{cases}$$

$$x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, x_4 \geq 0, x_5 \geq 0$$

$$x_1, x_2, x_3, x_4, x_5 \in Z$$

Решение. Находим решение без учета целочисленности задачи симплексным методом: $x^* = \left(\frac{18}{5}, \frac{3}{5}, 0, 0, \frac{24}{5}\right)$, $F(x^*) = \frac{39}{5}$.

Рассмотрим следующую пару задач:

$$\text{Задача 1: } F = 2x_1 + x_2 \to \max$$
$$\begin{cases} 6x_1 + 4x_2 + x_3 = 24 \\ 3x_1 - 3x_2 + x_4 = 9 \\ -x_1 + 3x_3 + x_5 = 3 \end{cases}$$
$$x_1 \le 3$$
$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0, x_4 \ge 0, x_5 \ge 0$$

и

$$\text{Задача 2: } F = 2x_1 + x_2 \to \max$$
$$\begin{cases} 6x_1 + 4x_2 + x_3 = 24 \\ 3x_1 - 3x_2 + x_4 = 9 \\ -x_1 + 3x_3 + x_5 = 3 \end{cases}$$
$$x_1 \ge 4$$
$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0, x_4 \ge 0, x_5 \ge 0.$$

Первая задача имеет оптимальный план $x_1^* = \left(3, \frac{3}{2}, 0, \frac{9}{2}, \frac{3}{2}\right)$, вторая – неразрешима.

Проверяем на целочисленность план первой задачи. Это условие не выполняется, поэтому строим следующие задачи:

$$\text{Задача 1.1: } F = 2x_1 + x_2 \to \max$$
$$\begin{cases} 6x_1 + 4x_2 + x_3 = 24 \\ 3x_1 - 3x_2 + x_4 = 9 \\ -x_1 + 3x_3 + x_5 = 3 \end{cases}$$
$$x_1 \le 3$$
$$x_2 \le 1$$
$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0, x_4 \ge 0, x_5 \ge 0$$

и

$$\text{Задача 1.2: } F = 2x_1 + x_2 \to \max$$
$$\begin{cases} 6x_1 + 4x_2 + x_3 = 24 \\ 3x_1 - 3x_2 + x_4 = 9 \\ -x_1 + 3x_3 + x_5 = 3 \end{cases}$$
$$x_1 \ge 4$$
$$x_2 \ge 2$$
$$x_1 \ge 0, x_2 \ge 0, x_3 \ge 0, x_4 \ge 0, x_5 \ge 0$$

Задача 1.2 неразрешима, а задача 1.1 имеет оптимальный план $x^*_{1.1} = (3, 1, 2, 3, 3)$, на котором значение целевой функции $F(x^*_{1.1}) = 7$.

В результате получили, что исходная задача целочисленного программирования имеет оптимальный план $x^*_{1.1} = (3, 1, 2, 3, 3)$ и $F(x^*_{1.1}) = 7$.

Главный недостаток алгоритма метода ветвей и границ заключается в необходимости полностью решать задачи линейного программирования, ассоциированные с каждой из вершин многогранника допустимых решений. Для задач большой размерности это требует значительных и, в известной степени, неоправданных с практической точки зрения затрат времени. Но, несмотря на недостатки данного метода, можно утверждать, что в настоящее время алгоритм метода ветвей и границ является наиболее надежным средством решения целочисленных задач, встречающихся в практических исследованиях. Это не означает, однако, что любая целочисленная задача может быть решена с помощью рассматриваемого метода.

Рассмотрим общую схему метода ветвей и границ.

Общая схема метода. Этот метод применим как к полностью целочисленным задачам, так и частично целочисленным задачам.

Рассмотрим задачу

$$c'x \to \max, \tag{18}$$

$$Ax \le b, \tag{19}$$

$$d_{*j} \le x_j \le d^*_j, \ j \in J, \tag{20}$$

$$x_j - \text{целое}, \ j \in I, \tag{21}$$

$$x = (x_j, j \in J), J = \{1, 2, \ldots, n\}, I \subset J.$$

Заметим, что без ограничения общности для $j \in I$ числа x_j можно считать целыми.

1.4. Метод решения задачи о назначениях

Задача о назначениях – задача о наилучшем распределении некоторого числа работ между таким же числом исполнителей. При ее решении ищут оптимальное назначение из условия максимума общей производительности, которая равна сумме производительности исполнителей.

В процессе управления производством зачастую возникают задачи назначения исполнителей на различные виды работ, например: подбор кадров и назначение кандидатов на вакантные должности, распределение источников капитальных вложении между различными проектами научно-технического развития, распределение экипажей самолетов между авиалиниями.

Задача о назначениях. Пусть имеется n исполнителей, которые могут выполнять n различных работ. Известна полезность c_{ij}, связанная с выполнением i-м исполнителем j-й работы, i =1, ..., n, j =1,...,n. Необходимо назначить исполнителей на работы так, чтобы добиться максимальной полезности, при условии, что каждый исполнитель может быть назначен только на одну работу и за каждой работой должен быть закреплен только один исполнитель.

Математическая модель задачи примет вид

$$F = \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \to \max \tag{18}$$

Каждый исполнитель назначается только на одну работу:

$$\sum_{j=1}^{n} x_{ij} = 1 \tag{19}$$

На каждую работу назначается только один исполнитель:

$$\sum_{i=1}^{n} x_{ij} = 1 \tag{20}$$

Условия неотрицательности и целочисленности:

$$x_{ij} \ge 0,\ x_{ij} \in \{0,1\},\ i = 1,\dots,n,\ j = 1,\dots,n. \tag{21}$$

При решении задачи о назначениях исходной информацией является таблица задачи о назначениях c={ c_{ij} }, элементами которой служат показатели эффективности назначений.

Для задачи о назначениях, записанной в стандартной форме, количество строк этой таблицы совпадает с количеством столбцов.

Таблица 1.1

Рабочий \ Работа	1	2	…	j	…	m
1	c_{11}	c_{12}	…	c_{1j}	…	c_{1m}
2	c_{21}	c_{22}	…	c_{2j}	…	c_{2m}
…	…	…	…	…	…	…
i	c_{i1}	c_{i2}	…	c_{ij}	…	c_{im}
…	…	…	…	…	…	…
m	c_{m1}	c_{m2}	…	c_{mj}	…	c_{mm}

Результатом решения задачи о назначениях (18) – (21) является вектор $x^* = \{x_{ij}^*\}$, компоненты которого – целые числа. Оптимальный план задачи о назначениях (18) – (21) можно представить в виде квадратной матрицы назначений, в каждой строке и в каждом столбце которой находится ровно одна единица.

Такую матрицу иногда называют матрицей перестановок. Значение целевой функции (18), соответствующее оптимальному плану, называют эффективностью назначений.

Если количество рабочих не равно количеству работ, то возникает задача о назначениях в открытой форме. В этом случае задача может быть преобразована в задачу, сформулированную в стандартной форме.

Пусть m – количество работ. И пусть, например, количество рабочих n превышает количество работ m. Введем дополнительные фиктивные работы с индексами $j = m+1, ..., n$. Коэффициенты таблицы назначений c_{ij}, $i = 1, ..., n$; $j=m+1, ..., n$, положим равными нулю. В этом случае получаем задачу, сформулированную в стандартной форме. Если в оптимальном плане этой задачи $x_{ij}^* =1$ при $j=m+1, ..., n$, то исполнитель i назначается на выполнение фиктивной работы, т.е. остается без работы.

Заметим, что оптимальное значение целевой функции исходной задачи совпадает с оптимальным значением задачи, приведенной к стандартной форме. Поэтому эффективность назначений в результате такого преобразования не меняется.

Следует особо отметить, что задача о назначениях является частным случаем транспортной задачи, в которой количество пунктов производства совпадает с количеством пунктов потребления, а все величины спроса и величины предложения равны.

Задача 1.4. Цех металлообработки получил срочный заказ на выпуск партии деталей. Для производства детали необходимо выполнить операции на четырех станках. В цехе работают четыре слесаря высокой квалификации, каждый из которых может работать на любом станке, но с различным процентом брака (процент брака известен из документации ОТК).

Таблица 1.2

Станок / Рабочий	1	2	3	4
1	2,3	1,9	2,2	2,7
2	1,8	2,2	2,0	1,8
3	2,5	2,0	2,2	3,0
4	2,0	2,4	2,4	2,8

Распределите станки между рабочими таким образом, чтобы процент брака был минимальным. (Предполагается, что ОТК проверяет готовую деталь, т.е. общий процент брака определяется как сумма процентов брака, допущенного всеми рабочими.)

Вопросы:

1. На каком станке должен работать рабочий 2?

2. Чему равен минимальный общий процент брака?

Решение. Необходимо провести оптимизационные расчеты. После этого получим матрицу назначений (таблица 1.3).

Таблица 1.3

Станок / Рабочий	1	2	3	4
1	0	1	0	0
2	0	0	0	1
3	0	0	1	0
4	1	0	0	0

Решение можно представить в виде таблицы 1.4.

Таблица 1.4

Рабочий	Станок	Процент брака
1	2	1,9
2	4	1,8
3	3	2,2
4	1	2,0
Итог		7,9

Ответ:

1. Рабочий 2 должен работать на станке № 4.

2. Минимальный общий процент брака равен 7,9%.

Задача 1.5. Фирма получила заказы на разработку пяти программных продуктов. На фирме работают шесть квалифицированных программистов, которым можно поручить выполнение этих заказов. Каждый из них дал оценку времени (в днях), требуемого для разработки программ. Эти оценки приведены в таблице 1.5.

Таблица 1.5

Программа / Программист	1	2	3	4	5
Галкин	46	59	24	62	67
Палкин	47	56	32	55	70
Малкин	44	52	19	61	60
Чалкин	47	59	17	64	73
Залкинд	43	65	20	60	75
Кузьмин	41	53	28	54	68

Выполнение каждого из пяти заказов фирма решила поручить одному программисту. Ясно, что один из программистов не получит заказа. Каждому программисту, которому будет поручено выполнять заказ, фирма предложила оплату 1 тыс.руб. в день. Распределите работу между программистами, чтобы общие издержки на разработку программ были минимальными.

Вопросы:

1. Чему равны минимальные издержки фирмы на выполнение всех пяти заказов?

2. Какую программу следует поручить Малкину?

3. Какую программу следует поручить Залкинду?

4. Кто из программистов не получит заказа?

5. Стало известным, что не все программисты согласились с условиями оплаты, обосновывая это тем, что имеют разную квалификацию. В результате была достигнута договоренность о следующих размерах оплаты в день (в тыс. руб.) (таблица 1.6). Изменится ли распределение работ между программистами при новых условиях оплаты труда? Каковы будут в этом случае общие минимальные издержки?

6. Кто из программистов при новых условиях не получит заказа?

Таблица 1.6

Программист	Размер оплаты
Галкин	1
Палкин	2
Малкин	1,5
Чалкин	2
Залкинд	1,5
Кузьмин	2

Решение. В таблице 1.7 задачи о назначениях указан размер оплаты (в тыс. руб.) в случае, если программисту будет поручена соответствующая программа.

Таблица 1.7

Программа / Программист	1	2	3	4	5	6 (фиктивная)
Галкин	46	59	24	62	67	0
Палкин	47	56	32	55	70	0
Малкин	44	52	19	61	60	0
Чалкин	47	59	17	64	73	0
Залкинд	43	65	20	60	75	0
Кузьмин	41	53	28	54	68	0

После проведения расчетов получается следующая матрица назначений (таблица 1.8).

Таблица 1.8

Программа / Программист	1	2	3	4	5	6 (фиктивная)
Галкин	0	0	0	0	0	0
Палкин	0	0	0	1	0	0
Малкин	0	0	0	0	1	0
Чалкин	0	0	1	0	0	0
Залкинд	1	0	0	0	0	0
Кузьмин	0	1	0	0	0	0

Учитывая исходную информацию, получается следующая таблица 1.9.

Таблица 1.9

Программист	Программа	Издержки, тыс. руб.
Галкин	Фиктивная	0
Палкин	4	55
Малкин	5	60
Чалкин	3	17
Залкинд	1	43
Кузьмин	2	53
Итог		228

Из последней таблицы следует, что минимальные издержки составляют 228 тыс. руб., Малкину следует поручить программу 5, а Залкинду – программу 1. Заказа не получит Галкин. При новых условиях оплаты труда таблица задачи о назначениях имеет следующий вид (таблица 1.10).

Таблица 1.10

Программа / Программист	1	2	3	4	5	6 (фиктивная)
Галкин	46	59	24	62	67	0
Палкин	94	112	64	110	140	0
Малкин	66	78	28,5	91,5	90	0
Чалкин	94	118	34	128	146	0
Залкинд	64,5	97,5	30	90	112,5	0
Кузьмин	82	106	56	108	136	0

После проведения расчетов, получается следующая матрица назначений (таблица 1.11).

Таблица 1.11

Программист \ Программа	1	2	3	4	5	6 (фиктивная)
Галкин	0	0	0	0	1	0
Палкин	0	0	0	0	0	0
Малкин	0	1	0	0	0	0
Чалкин	0	0	1	0	0	0
Залкинд	0	0	0	1	0	0
Кузьмин	1	0	0	0	0	0

В таблице 1.12 представлен итоговый результат.

Таблица 1.12

Программист	Программа	Издержки, тыс. руб.
Галкин	5	67
Палкин	Фиктивная	0
Малкин	2	78
Чалкин	3	34
Залкинд	4	90
Кузьмин	1	82
Итог		351

Ответ:

1. Минимальные издержки фирмы на выполнение всех пяти заказов = 228 тыс. руб.

2. Следует поручить Малкину Программу 5.

3.Следует поручить Залкинду Программу 1.

4. Не получит заказ программист Галкин.

5. Общие минимальные издержки = 351 тыс. руб.

6. При новых условиях не получит заказа программист Палкин.

1.5. Метод Гомори для полностью целочисленных задач

1. Интуитивное обоснование метода. Отметим ещё одну особенность задач ЦЛП, которая отличает их от соответствующих задач ЛП. Рассмотрим задачу:

$$c'x \to \max, \tag{22}$$

$$Ax = b, \ \ x \geq 0, \tag{23}$$

$$x_j - \text{целое}, \ \ j \in J,$$

где $A \in R^{m \times n}$, $\operatorname{rank} A = m < n$, $J = \{1, 2, ..., n\}$.

Если мы рассмотрим задачу ЛП (22), то увидим, что среди оптимальных планов задачи (22) всегда существует базисный план, т. е. план, у которого не более чем m компонент, отличных от нуля.

Рассмотрим теперь задачу ЦЛП (22), (23) с m основными ограничениями. В общем случае оптимальный план задачи (22), (23) будет содержать более чем m компонент, отличных от нуля. Это сразу же наводит на мысль о том, что если мы хотим получить решение целочисленной задачи (22), (23) как решение некоторой «непрерывной» задачи ЛП, то эта «непрерывная» задача должна содержать более чем m основных ограничений, т.е. мы должны к задаче (22) добавить еще некоторые основные ограничения.

Необходимость введения новых ограничений в задачу (22) для получения решения задачи (22), (23) можно объяснить и с другой позиции. Рассмотрим множество допустимых планов X задачи (22) (рис.1.3). Понятно, что множеством допустимых планов задачи Хц (22), (23) будут все «целые» точки, принадлежащие X. Рассмотрим ещё одно множество Хв, называемое выпуклой оболочкой точек . Согласно определению, это наименьшее выпуклое множество, содержащее все точки X (см. рис. 1.3). Справедливы включения

$$X_ц \subset X_В \subset X. \tag{24}$$

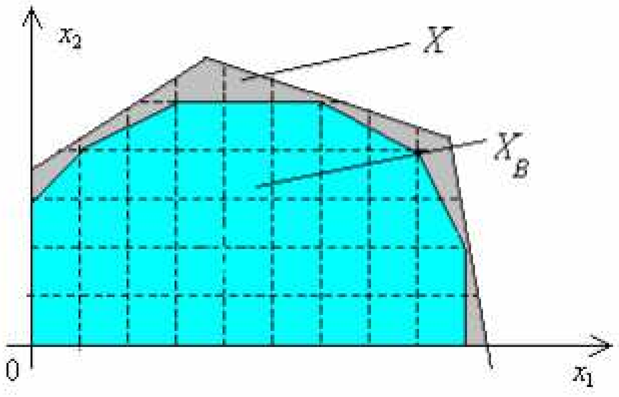


Рис. 1.3

Множество Хц, так же как и множество X, является «непрерывным». Легко видеть, что все угловые точки множества Хв – целые точки. Отсюда, учитывая свойства симплекс-метода и включения (24), заключаем, что среди решений задачи ЛП

$$c'x \to max, x \in X_B \qquad (25)$$

обязательно есть целочисленное решение и это решение будет решением задачи (22), (23). Мы видим, что множество Хв получилось из X путём «отсечения» лишних кусков, не содержащих целых точек. При этом важно, что мы не отсекли ни одной целой точки. На рисунке для R2 построить множество Хв просто. При большом n сделать это заранее перед решением задачи невозможно. Кроме того, нам не надо всё «чистое» множество Хв. Нам важно «высечь» только часть этого множества в окрестности оптимальной точки.

На этих идеях отсечений и основан метод Гомори. Суть метода состоит в том, что на каждом шаге к ограничениям соответствующей задачи ЛП добавляется новое ограничение, называемое отсекающей плоскостью. Эта плоскость должна обладать следующими свойствами:

1) отсекать имеющееся в наличии нецелочисленное решение задачи ЛП;

2) не отсекать ни одного целочисленного плана исходной задачи ЦЛП;

3) отсечения должны строиться таким образом, чтобы обеспечить конечность алгоритма, т.е. решение задачи ЦЛП должно строиться за конечное число шагов.

2. Построение отсекающей плоскости. Рассмотрим задачу ЦЛП вида (22), (23). Предположим, что:

1) ограничения задачи (22), (23) совместны;

2) целевая функция ограничена сверху.

Прежде чем описать весь алгоритм, покажем в общем случае, как построить дополнительное ограничение, которому удовлетворяют все целые планы задачи (22), (23)

Рассмотрим любое линейное уравнение, которое можно получить из уравнений Ax = b путём алгебраических операций над этими уравнениями. В общем случае такое уравнение можно представить в виде

$$y' Ax = y' b, \tag{26}$$

где $y \in R^m$, $y \neq 0$, – произвольный вектор.

Очевидно, из условия Ax = b следует справедливость (26). Обозначим

$a_j = y' A_j,\ j \in J,\ \beta = y' b.$

Тогда равенство (26) примет вид

$$\sum_{j=1}^{n} a_j x_j = \beta . \tag{27}$$

Предположим, что хотя бы одно из чисел x_j, a_j ($j \in J$), β является дробным. Как и раньше, обозначим через [d] целую часть числа d ([d] – наибольшее целое, не превосходящее d). По построению любой вектор x, удовлетворяющий Ax = b, должен удовлетворять и (27). Так как все $x_j \geq 0$, $j \in J$ и $[a_j] \leq a_j$, $j \in J$, то из (27) следует более слабое условие

$$\sum_{j=1}^{n} [a_j] x_j \leq \beta . \tag{28}$$

Теперь учтём, что по условию все x_j, $j \in J$ – целые. Отсюда следует, что все целые x_j, $j \in J$, удовлетворяющие (28), должны удовлетворять более сильному, чем (28), неравенству

$$\sum_{j=1}^{n}[a_j]x_j \le [\beta]. \tag{29}$$

Заметим, что в общем случае, без учёта целочисленности, из (28) не следует (29).

Условие (29) можно переписать в эквивалентной форме

$$\sum_{j=1}^{n}[a_j]x_j + x_* = [\beta], \;\; x_* \ge 0, \;\; x_* - \text{целое}. \tag{30}$$

Таким образом, условие (30) есть новое условие, которому удовлетворяют все целые планы исходной задачи (22) – (23). В алгоритме используется не условие (30), а разность между (30) и (27). Определим значения f_j и f тождественными

$$[a_j] + f_j \equiv a_j, \;\; j \in J, \;\; [\beta] + f = \beta,$$

т.е. f_j – дробная часть числа a_j , $j \in J$, а f – дробная часть числа β . По построению,

$$0 \le f_j < 1, \;\; j \in J, \;\; 0 \le f < 1.$$

Вычтем из (30) равенство (27), в результате получим

$$\sum_{j=1}^{n}[-f_j]x_j + x_* = -f, \;\; x_* \ge 0 - \text{целое}. \tag{31}$$

В алгоритме вместо (30) используется отсекающее ограничение (31).

3. Алгоритм Гомори (общая схема). Общую схему алгоритма Гомори можно представить в виде последовательности следующих шагов.

Шаг 1. Найти оптимальное решение задачи линейного программирования.

Шаг 2. Если в оптимальном базисе задачи ЛП есть искусственные индексы и размеры задачи ЛП велики, то уменьшаем размеры текущей задачи ЛП: удаляем из задачи ЛП искусственные переменные с базисными индексами и соответствующие им ограничения. В противном случае сразу переходим к шагу 3 без уменьшения размеров задачи ЛП.

Шаг 3. Прекращаем решение задачи ЦЛП, если все неискусственные переменные задачи ЛП целые. В противном случае переходим к шагу 4.

Шаг 4. Сформируем отсекающую плоскость (ограничение). Для этого выберем любую дробную неискусственную переменную (это обязательно базисная переменная) $x^0_{i_0}$, $i_0 \in J^0_Б$.

Здесь $J^0_Б$ – оптимальный базис текущей задачи ЛП. Положим

$$y' = e'_{i_0} (A^0_Б)^{-1}.$$

Сформируем отсекающее ограничение (31), исходя из ограничения (27), построенного по правилам предыдущего пункта.

Шаг 5. Добавляем отсекающее ограничение (41) и новую (целую!) переменную к задаче ЛП и получаем расширенную (новую) задачу ЛП. Переходим к шагу 1.

Опишем перечисленные шаги алгоритма подробнее. Рассмотрим шаг 1. Пусть в текущей задаче ЛП есть переменные

$$x_j \geq 0, \; j \in J \cup J_{иск},$$

и есть $\overline{m} \geq m$ основных ограничений. По построению

$$\overline{m} = m + |J_{иск}|$$

и каждой искусственной переменной xj поставлено в соответствие некоторое основное ограничение текущей задачи ЛП. Отметим, что это ограничение не входит в число основных ограничений исходной задачи ЛП или ЦЛП, т.е. является дополнительным отсекающим ограничением. Решим текущую задачу ЛП и обозначим через

$$x^0_j, \; j \in J \cup J_{иск}, \;\; J^0_Б \subset J \cup J_{иск}, \;\; |J^0_Б| = \overline{m}$$

ее оптимальный базисный план.

Опустим пока подробное описание шага 2, поскольку на первой итерации этот шаг «пропускается» и поэтому его удобнее описывать после описания всех других шагов.

Шаг 3 описан подробно ранее.

Рассмотрим шаг 4. Предположим, что выбран некоторый индекс i_0, такой, что

$i_0 \in J_Б^0 \cap J,\ x_{i_0}^0 > 0$ – нецелое.

Положим

$$y' = e'_{i_0}(A_Б^0)^{-1},\ a_j = y'A_j,\ j \in J \cup J_{иск},\ \beta = y'b,$$

где Aj – j-й столбец матрицы условий и b – вектор правых частей основных ограничений текущей задачи ЛП. Рассмотрим ограничение

$$\sum_{j \in J \cup J_{иск}} a_j x_j = \beta. \quad (32)$$

Так как по построению, в силу специального выбора вектора y, имеют место равенства

$$a_{i_0} = 1,\ a_j = 0,\ j \in J_Б \setminus i_0,$$

то равенство (32) можно представить в виде

$$x_{i_0} + \sum_{j \in J_{\text{Н}}} a_j x_j = \beta, \quad (33)$$

где $J_{\text{Н}} \subset (J \cup J_{иск}) \setminus J_Б^{\,0}$. Так как все $x_j^{\,0} = 0,\ j \in J_{\text{Н}}$, и вектор $x^0 = (x_j^0,\ j \in$

$\in J \cup J_{иск})$

удовлетворяет (33), то имеем

$\beta = x_{i_0}^{\,0} > 0$ – нецелое.

Отсекающее ограничение (31), построенное по (33), имеет вид

$$\sum_{j \in J_{\text{Н}}} [-f_j] x_j + x_{j*} = -f,\ \ x_{j*} \geq 0 \text{ – целое}, \quad (34)$$

где x_{j*} – новая переменная, соответствующая отсекающему ограничению (34), j* – наименьший целый индекс, не принадлежащий множеству $J \cup J_{иск}$.

Рассмотрим шаг 5. Ограничение (34) добавляем к основным ограничениям имеющейся задачи ЛП, индекс j* и соответствующую ему переменную x_{j*} добавляем к переменным текущей задачи ЛП. При этом произойдут следующие замены:

$$\overline{m} \to \overline{m}+1;\ J \cup J_{иск} \to J \cup \bar{J}_{иск},\ \bar{J}_{иск} = J_{иск} \cup j..$$

Покажем, что добавляемое новое ограничение (34) является «отсекающим», т.е. оно отсекает нецелочисленное оптимальное решение «старой» задачи ЛП. Действительно, на оптимальном плане x0 «старой» задачи ЛП имеют место соотношения:

$$x_j^{\,0} = 0,\ j \in J_{\text{Н}},$$

и по построению, 0< f<1. Следовательно, для выполнения ограничения (34) надо положить

$$x_{j*}^{0} = -f < 0.$$

Однако это противоречит условию

$$x_{j*}^{0} \geq 0.$$

Таким образом, мы видим, что оптимальный план «старой» задачи ЛП не удовлетворяет ограничению (44) и, следовательно, не является планом новой задачи ЛП. В этой ситуации новую задачу ЛП разумно решать двойственным симплекс-методом [2, 8], начиная процесс вычисления с оптимального двойственного базисного плана старой задачи ЛП.

Рассмотри шаг 2. Для того чтобы размеры задачи ЛП не росли неограниченно, поступаем следующи образом. Если $J_Б^0 \cap J_{иск} \neq 0$, то перед тем, как построить новое отсекающее ограничение, выбросим отсекающее ограничение, породившую искусственную переменную x_{j*}, $j^* \in J_Б^0 \cap J_{иск}$. Исключим из задачи ЛП искусственную переменную x_{j*}, подставив вместо неё в другие отсекающие ограничения, содержащие переменную x_j, её выражение через другие переменные, найденное из «выбрасываемого» отсекающего ограничения.

Дополнительное ограничение, соответствующее x_{j*}, можно выбросить, так как вхождение искусственной переменной x_{j*} в оптимальный базис означает, что это ограничение пассивно. Используя эту процедуру выбрасывания лишних ограничений, всегда можно добиться того, что в решаемой задаче ЛП будет не более чем n основных ограничений.

Теоретически доказано, что описанный алгоритм сходится к оптимуму за конечное число итераций.

Пример. Рассмотрим задачу

$$21x_1 + 11x_2 \to \max, \tag{35}$$

$$7x_1 + 4x_2 + x_3 = 13, \quad x_j \geq 0, \quad j = \overline{1,3}, \tag{36}$$

$$x_j \geq 0 - \text{целое}, \quad j = \overline{1,3}. \tag{37}$$

Решим данную задачу методом Гомори. Алгоритм опишем по итерациям.

Итерация 1. Решим задачу (35), (36) симплекс-методом. В результате получим оптимальный базисный план

$$x_1^0 = \frac{13}{7}, \ x_2^0 = x_3^0 = 0, \ J_Б^0 = \{1\}.$$

Базисная матрица $A_Б = (A_j, \ j \in J_Б^0)$ и обратная к ней матрица $A_Б^{-1}$ имеют вид

$$A_Б = 7, \ A_Б^{-1} = \frac{1}{7}.$$

Выберем i_0=1, подсчитаем вектор $y' = e_{i_0}' A_Б^{-1} = \frac{1}{7}$ и получим ограничение вида (27)

$$x_1 + \frac{4}{7}x_2 + \frac{1}{7}x_3 = 1\frac{6}{7}.$$

По последнему ограничению построим отсекающее ограничение вида (31)

$$-\frac{4}{7}x_2 - \frac{1}{7}x_3 + x_4 = -\frac{6}{7}, \ x_4 \geq 0. \tag{38}$$

Добавим ограничение (38) к задаче (35), (36) и перейдем к следующей итерации.

Итерация 2. Решим задачу ЛП (35), (36), (38). Она имеет решение:

$$x_1^0 = 1, \ x_2^0 = 1\frac{1}{2}, \ x_3^0 = 0, \ x_4{}^0 = 0, \ J_Б^0 = \{1, 2\}.$$

Базисная матрица $A_Б = (A_j, \ j \in J_Б^0)$ и обратная к ней матрица $A_Б^{-1}$ имеют вид

$$A_Б = \begin{pmatrix} 7 & 4 \\ 0 & -\frac{4}{7} \end{pmatrix}, \ A_Б{}^{-1} = \begin{pmatrix} \frac{1}{7} & 1 \\ 0 & -\frac{7}{4} \end{pmatrix}.$$

Выберем i_0=2, положим $y' = (0, \quad 1)A_Б^{-1} = (0, -\frac{7}{4})$. Используя найденный вектор у, построим ограничение вида

$$\sum_{j \in J_H} a_j x_j = \beta, \qquad (39)$$

где $a_j = y'A_j$, $\beta = y'b$. На данной итерации матрица условий A и вектор правых частей b текущей задачи ЛП имеют вид

$$A=\begin{pmatrix} 7 & 4 & 1 & 0 \\ 0 & -\frac{4}{7} & -\frac{1}{7} & 1 \end{pmatrix}, \quad b=\begin{pmatrix} 13 \\ -\frac{6}{7} \end{pmatrix}.$$

Следовательно, a_1=0, a_2=1, a_3=1/4, a_4=-7/4, β=3/2 и ограничение (39) принимает вид

$$x_2 + \frac{1}{4} x_3 - \frac{7}{4} x_4 = 3/2.$$

По этому ограничению сформируем отсекающее ограничение вида (31)

$$-\frac{1}{4} x_3 - \frac{1}{4} x_4 + x_5 = -\frac{1}{2}. \qquad (40)$$

Добавим ограничение (40) к задаче (35), (36), (38) и перейдем к следующей итерации.

Итерация 3. Решим задачу ЛП (35), (36), (38), (40) симплекс-методом. Она имеет решение:

$$x_1^0=0, \; x_2^0=3, \; x_3^0=1, \; x_4^0=1, \; x_5^0=0, \; J_Б^0=\{2, 3, 4\}.$$

Это решение является целочисленным. Следовательно, оно является решением исходной задачи ЦЛП (35)-(37).

Отметим, что если бы последнее решение было нецелочисленным, то на следующей итерации перед началом формирования нового отсекающего ограничения нужно было бы удалить из последней задачи ЛП (т.е. из задачи (35), (36), (38), (40)) ограничение (38), «породившее» искусственную переменную x_4, которая вошла в оптимальный базис, а в остальные ограничения (например, в (40)) подставили бы вместо переменной x_4 ее выражение через другие переменные, найденное из (38), т.е. в (40) вместо x_4 подставили бы

$$x_4 = \frac{4}{7}x_2 + \frac{1}{7}x_3 - \frac{6}{7}.$$

В результате ограничение (40) приняло бы вид

$$-\frac{1}{7}x_2 - \frac{2}{7}x_3 = -\frac{5}{7} \text{ или } x_2 + 2x_3 = 5.$$

Рассмотрим несколько подходов к алгоритму метода Гомори.

МЕТОД ГОМОРИ-1

Постановка целочисленной задачи линейного программирования

Найти вектор $\boldsymbol{x} = (x_1 \dots, x_n)$, что минимизирует целевую функцию

$$L(x) = c_1x_1 + \dots + c_nx_n \tag{41}$$

и удовлетворяет систему ограничений

$$a_{11}x_1 + \dots + a_{1n}x_n = a_{10}$$

$$\dots\dots\dots\dots\dots\dots \tag{42}$$

$$a_{m1}x1 + \dots + a_{mn}x_n = a_{m0}$$

$$x_j \geq 0,\ j = 1 \dots, n \tag{43}$$

x_j — целые, $j = 1 \dots, n$. (44)

Изложение метода Гомори-1

Метод Гомори-1 является одним из методов отсечения, идея которых заключается в следующем.

Решается вспомогательная *ЗЛП* (41) – (43), которую получают из исходной *ЦЗЛП* (41) – (44) отбрасыванием условия целочисленности переменных (44). Если оптимальное решение вспомогательной *ЗЛП* – целочисленное, то оно будет и решением исходной *ЦЗЛП*. Если же полученное решение вспомогательной задачи не является целочисленным, то от развязанной *ЗЛП* переходят к новой вспомогательной *ЗЛП* присоединением линейного ограничения, которое удовлетворяет целочисленности исходной *ЦЗЛП* и которое не удовлетворяет полученному нецелочисленному решению вспомогательной *ЗЛП*. Упомянутое дополнительное линейное ограничение определяет некоторую отрезающую плоскость и называется *правильным ограничением*. Присоединение новых правильных ограничений к начальной вспомогательной *ЗЛП* осуществляется до тех пор, пока на некотором шаге не будет получено целочисленное решение вспомогательной задачи, которое, очевидно, будет оптимальным решением исходной *ЦЗЛП*. В методе Гомори-1 правильное *ограничение* строится таким образом.

Пусть на последней итерации симплекс-метода при решении вспомогательной *ЗЛП* непрямые ограничения этой задачи приобрели вид

$$x_i + a'_{i,m+1} x_{m+1} + \ldots + a'_{in} x_n = a'_{i0}, \quad i=1\ldots,m$$

и, таким образом, решением вспомогательной *ЗЛП* будет вектор

$$\mathbf{x} = (a'_{10}\ldots, a'_{m0}, 0\ldots, 0).$$

Пусть существует номер r такой, что a'_{r0} − дробь и $\{z\}$ − дробная часть z. Тогда правильное ограничение по методу Гомори-1 задается неравенством

$$\{a'_{r,m+1}\} x_{m+1} + \ldots + \{a'_{rn}\} x_n \geq \{a'_{r0}\}.$$

<u>Алгоритм метода Гомори-1</u>

1. Решаем вспомогательную *ЗЛП* (41) − (4.3). Пусть $\boldsymbol{x(0)}$ − ее оптимальное решение. Если оптимальное решение не существует, то исходная *ЦЗЛП* также не имеет оптимального решения.

2. Пусть на s-*й* итерации задана вспомогательная *ЗЛП*, которая имеет M ограничений и N переменных, $\boldsymbol{x(s)}$ − ее оптимальное решение.

Будем считать, что $\boldsymbol{x(s)}$ определяется каноничными ограничениями последней итерации, то есть

$$x_i + b_{i,M+1} x_{M+1} + \ldots + b_{iN} x_N = b_{i0}, \quad i=1\ldots,M.$$

Откуда

$$\boldsymbol{x(s)} = (b_{10}\ldots, b_{M0}, 0, \ldots, 0).$$

3. Если b_{i0} $(i=1\ldots,M)$ – целые, то – конец алгоритма $\boldsymbol{x(s)}$ является оптимальным решением исходной *ЦЗЛП*. Если существует хотя бы одно такое решение, что b_{i0} – дробь, то переход к пункту 4.

4. Находим $r = min\{\, i\,\}$ по всем i таким, что b_{i0} – дробь, и строим дополнительное ограничение

$$x_{N+1} - \{b_{r,M+1}\}\, x_{M+1} - \ldots - \{b_{rN}\}\, x_N = -\{b_{r0}\}\,,$$

где $x_{N+1} \geq 0$ – дополнительная переменная.

5. Расширяем симплекс-таблицу за счет $(M+1)$ -ой строки (дополнительное ограничение) и (N+1) -го столбца, *что* отвечает дополнительной переменной x_{N+1}.

6. Решаем расширенную таким образом *ЗЛП* двойственным симплекс-методом и переходим к пункту 2 с заменой s на $s+1$. Если при этом на какой-либо итерации двойственного симплекс-метода одна из дополнительных переменных задачи повторно становится базисной, то исключаются из последующего рассмотрения соответствующие ей строка и столбец.

МЕТОД ГОМОРИ-2

<u>Постановка частично целочисленной задачи линейного программирования (ЧЦЗЛП)</u>

Найти вектор $\boldsymbol{x} = (x_1\ldots,x_n)$, который минимизирует целевую функцию

$$L(x) = c_1x_1 + \ldots + c_nx_n \qquad (45)$$

и удовлетворяет систему ограничений

$$\begin{aligned} &a_{11}x_1 + \ldots + a_{1n}\, x_n = a_{10} \\ &\ldots\ldots\ldots\ldots\ldots\ldots \qquad (46) \\ &a_{m1}x_1 + \ldots + a_{mn}x_n = a_{m0} \end{aligned}$$

$$x_j \geq 0,\ j=1\ldots,n \qquad (47)$$

x_j — целые, $j=1\ldots,p$ $(p \leq n)$. (48)

<u>Изложение метода Гомори-2</u>

Метод Гомори-2, как и метод Гомори-1, является одним из методов отсечения и заключается в следующем.

Решается вспомогательная *ЗЛП* (45) – (47), которую получают из исходной ЗЛП (45) – (48) отбрасыванием условия целочисленности переменных (48). Если ее решение удовлетворяет условию (48), то оно же является и решением исходной *ЧЦЗЛП*. Иначе от решения ЗЛП переходят к новой вспомогательной *ЗЛП* присоединением линейного ограничения, которому удовлетворяют целочисленные (в понимании условий (48)) развязки исходной ЧЦЗЛП, но не удовлетворяет полученное нецелочисленное решение исходной ЗЛП. Упомянутое дополнительное ограничение определяет некоторую отрезающую плоскость и называется правильным ограничением. *Присоединение новых* правильных ограничений осуществляется до тех пор, пока на некотором шаге не будет получено целочисленное (в понимании условий (48)) решение вспомогательной задачи, которое и является оптимальным решением исходной ЧЦЗЛП. В методе Гомори-2 правильное ограничение строится так.

Пусть на последней итерации симплекс-метода при решении вспомогательной *ЗЛП* ее непрямые ограничения приобрели вид

$$x_i + Q_{i,m+1}\, x_{m+1} + \ldots + Q_{in}\, x_n = Q_{i0},\ i=1\ldots,m$$

и, значит, решением вспомогательной *ЗЛП* является вектор

$$\boldsymbol{x} = (\, Q_{10}\ldots, Q_{m0}, 0, \ldots, 0\,).$$

Пусть существует номер r $(r \le p)$ такой, что Q_{r0} – нецелое, и $\{z\}$ – дробная часть z. Тогда правильное ограничение метода Гомори-2 имеет вид

$$x_{n+1} - D_{r,m+1} x_{m+1} - \ldots - D_{rn}\, x_n = -\{Q_{r0}\} \qquad (49)$$

где $x_{n+1} \ge 0$ – дополнительная переменная, и

$$D_{rj} = \begin{cases} \{Q_{rj}\}, & \text{при} \quad j \le p, \{Q_{rj}\} \le \{Q_{r0}\}, \\ \{Q_{r0}\}(1-\{Q_{rj}\})/(1-\{Q_{r0}\}), & \text{при} \quad j \le p, \{Q_{rj}\} > \{Q_{r0}\}, \\ Q_{rj}, & \text{при} \quad j > p, Q_{rj} \ge 0, \\ \{Q_{r0}\}(-Q_{rj})/(1-\{Q_{r0}\}), & \text{при} \quad j > p, Q_{rj} < 0. \end{cases} \qquad (50)$$

Алгоритм метода Гомори-2

1. Решаем вспомогательную ЗЛП (45) – (47). Пусть $\boldsymbol{x(0)}$ – ее оптимальное решение. Если эта задача не имеет решения, то исходная *ЧЦЗЛП* также не имеет решения.

2. Пусть на *s-й* итерации решена вспомогательная *ЗЛП*, которая имеет M ограничений и N переменных, $\boldsymbol{x(s)}$ – ее оптимальное решение. Допустим, что $\boldsymbol{x(s)}$ определяется каноничными ограничениями последней итерации, а именно:

$$x_i + Q_{i,M+1}\, x_{M+1} + \ldots + Q_{iN}\, x_N = Q_{i0},\ i=1\ldots,M,$$

откуда следует, что

$$\boldsymbol{x(s)} = (\ Q_{10}\ldots,Q_{M0},0,\ldots,0\ ).$$

3. Если Q_{i0} $(i=1\ldots,p)$ – целые, то конец алгоритма $\boldsymbol{x(s)}$ является оптимальным решением исходной *ЧЦЗЛП*. Если существует хотя бы одно такое решение, что Q_{i0} – нецелое $(i=1\ldots,p)$, то переход к пункту 4.

4. Находим $r = min\{\, i\, \}$ по всем i $(i=1\ldots,p)$ *таким,* что Q_{i0} – нецелое, и строим дополнительное ограничение по формулам (49)–(50) при $m = M$ и $n = N$.

5. Расширяем симплекс-таблицу за счет $(M+1)$ -ой строки (дополнительное ограничение) и (N+1) -го столбца, *что* отвечает дополнительной переменной x_{N+1}.

6. Решаем расширенную *ЗЛП* с помощью двойственного симплекс-метода (*ДСМ*) и переходим к пункту 2 с заменой s на $s+1$, M на $M+1$, N на $N+1$. Если на некоторой итерации *ДСМ* одна из дополнительных переменных задачи опять становится базисной, то из последующего рассмотрения исключаются соответствующие ей строка и столбец и при переходе к пункту 2 заменяется лишь s на $s+1$.

МЕТОД ГОМОРИ-3

Постановка целочисленной задачи линейного программирования (ЦЗЛП)

Найти вектор $\boldsymbol{x} = (x_1\ldots,x_n)$, что минимизирует целевую функцию

$L(x) = c_1 x_1 + ... + c_n x_n$ (51)

и удовлетворяет систему ограничений

$$a_{11} x_1 + ... + a_{1n} x_n = a_{10}$$

$$. . . . . \quad (52)$$

$$a_{m1} x1 + ... + a_{mn} x_n = a_{m0}$$

$x_j \geq 0, j = 1 \ldots, n$ (53)

x_j – целые, $j = 1 \ldots, n$. (54)

<u>Изложение метода Гомори-3</u>

Метод Гомори-3 заключается в следующем.

Пусть ограничения (52) *ЗЛП* (51)–(53) приведены к почти каноничному виду

$$x_i + \alpha_{i,m+1} x_{m+1} + \ldots + \alpha_{in} x_n = \alpha_{i0}, \; i = 1 \ldots, m \quad (55)$$

где α_{ij}, $i = 1 ..., m$, $j = 0, m+1 \ldots, n$ – целые и симплекс - разницы $\Delta_j \geq 0, j = 1 \ldots, n$.

Если $\alpha_{i0} \geq 0$, $i = 1 \ldots, m$, то ограничения (55) определяют оптимальное решение *ЦЗЛП* (51) – (54), иначе определяется некоторое целочисленное почти допустимое базисное решение (*МДБР*) исходной *ЗЛП*. Можно бы было, конечно, выбрать один из индексов i, для которого $\alpha_{i0} < 0$, и выполнить итерацию двойственного симплекс-метода. Однако в этом случае целочисленность новых параметров была бы, вообще говоря, нарушена через необходимость деления на ведущий элемент. Этого можно избежать лишь тогда, когда ведущий элемент равняется -1. Оказывается, что можно построить дополнительное ограничение, которому удовлетворяют все целочисленные развязки *ЦЗЛП* и которое вместе с тем определяет ведущую строку, которая имеет ведущий элемент -1. Строится оно за *i-м* ограничением системы (55), для которого α_{i0} – минимальное среди отрицательных α_{i0}, и имеет вид

$$x_{n+1} + \alpha_{m+1,m+1} x_{m+1} + \ldots + \alpha_{m+1,n} x_n = \alpha_{m+1,0},$$

где $x_{n+1} \geq 0$ – дополнительная переменная.

$$\alpha_{m+1,j} = [\alpha_{ij} / \alpha], j = 0, m+1 \ldots, n$$

$\alpha = \max\{-\alpha_{lj}\}, j = m+1\ldots,n.$

Следовательно, имеющаяся симплекс-таблица расширяется за счет $(m+1)$ -ой строки с элементами $\alpha_{m+1,j}$ ($\alpha_{m+1,j} = 0$ при $j = 1\ldots,m$), но единичного столбца $\boldsymbol{A}_{n+1}$, что отвечает дополнительной переменной x_{n+1}. Потом выполняется итерация двойственного симплекс-метода с $(m+1)$ -й ведущей строкой.

<u>Алгоритм метода Гомори-3</u>

1. Приводим исходную *ЗЛП* (51) – (53) к почти каноничной *ЗЛП* (*МКЗЛП*) с целочисленными коэффициентами, что определяет целочисленный *МДБР* $\boldsymbol{x(0)}$, для которого $\Delta_j \geq 0, j = 1\ldots,n$.

2. Пусть на *s-й* итерации полученная полностью целочисленная *МКЗЛП*

$$x_i \;\; + \alpha_{i,M+1}\, x_{M+1} + \ldots + \alpha_{iN}\, x_N = \alpha_{i0},\;\; i = 1\ldots,M,$$

что определяет целочисленный *N-мерный МДБР* $\boldsymbol{x(s)} = (\alpha_{10}..\alpha_{M0}, 0..,0)$.

3. Если $\alpha_{i0} \geq 0,\; i = 1\ldots,M,$ то конец алгоритма: $\boldsymbol{x(s)}$ – оптимальное решение *ЦЗЛП*. Иначе – переход к 4.

4. Если для некоторого i такого, что $\alpha_{i0} < 0$ $\alpha_{ij} \geq 0, j = 1\ldots,N$, то конец: исходная *ЦЗЛП* не имеет допустимых решений. Если таких i нет, то – переход к 5.

5. Находим $i = argmin\{\alpha_{i0}\}$, *где* i: $\alpha_{i0} < 0$. *Определяем* $\alpha = max\{-\alpha_{lj}\},\; j = M+1\ldots,N,$ *и строим дополнительное ограничение*

$$x_{N+1} + \alpha_{M+1,M+1}\, x_M + 1 + \ldots + \alpha_{M+1,N}\, x_N = \alpha_{M+1,0},$$

где $x_{N+1} \geq 0$ – дополнительная переменная, $\alpha_{M+1,j}$ – целая часть отношения $\alpha_{lj}/\alpha, \quad j = 0, M+1\ldots,N.$

6. Расширяем симплекс-таблицу за счет $(M+1)$ -ой строки (дополнительное ограничение) и (N+1) -го столбца, *что* отвечает дополнительной переменной x_{N+1}.

Пример решения задачи целочисленного программирования методом Гомори в пакете Mathcad

Решить задачу $F = 2X_1 + 4X_2 \to \max$ при ограничениях:

$$\begin{cases} 2x_1 + x_2 \le 19/3, \\ x_1 + 3x_2 \le 10, \end{cases}$$

$x_1, x_2 \ge 0$ – целые числа.

Решение. Эту задачу без учета целочисленности переменных обозначим задачей 1. Решая ее в любом математическом пакете (пример решения задачи линейного программирования в Mathcad приведен в ПРИЛОЖЕНИИ к данному примеру на рис. 1.6), получим оптимальное решение:

F1=14,533 при x1=1,8 и x2= 2,733.

Видно, что переменная x1=1,8 не удовлетворяет условию целочисленности, поэтому составим следующие две задачи (задача 2 и задача 3) с дополнительными граничными условиями, исключающими возможность получения x1=1,8 .

Задача 2

$F = 2x_1 + 4x_2 \to \max$

при ограничениях:

$$\begin{cases} 2x_1 + x_2 \le 19/3, \\ x_1 + 3x_2 \le 10, \end{cases}$$

$0 \le x_1 \le 1, \quad x_2 \ge 0,$

x_1, x_2 – целые числа.

Задача 3

$F = 2x_1 + 4x_2 \to \max$

при ограничениях:

$$\begin{cases} 2x_1 + x_2 \le 19/3, \\ x_1 + 3x_2 \le 10, \end{cases}$$

$x_1 \ge 2, \quad x_2 \ge 0,$

x_1, x_2 – целые числа.

При решении задач 2 и 3 получили следующие результаты: F2=13,999 при x1=1,0, x2= 2,999 и F3=13,28 при x1=2,0, x2=2,321. В обеих задачах нарушается условие целочисленности для переменной x2, поэтому вводим новые ограничения на граничные условия переменной x2. Для задачи 2 составляем две новые задачи – задачи 4 и 5 (исключая значения 2 < x2 < 3).

Задача 4

$F = 2x_1 + 4x_2 \to \max$

при ограничениях:

$$\begin{cases} 2x_1 + x_2 \le 19/3, \\ x_1 + 3x_2 \le 10, \end{cases}$$

$0 \le x_1 \le 1,\ 0 \le x_2 \le 2,$

x_1, x_2 – целые числа.

Задача 5

$F = 2x_1 + 4x_2 \to \max$

при ограничениях:

$$\begin{cases} 2x_1 + x_2 \le 19/3, \\ x_1 + 3x_2 \le 10, \end{cases}$$

$0 \le x_1 \le 1,\ x_2 \ge 3,$

x_1, x_2 – целые числа.

При решении задач 4 и 5 получили следующие результаты: F4=10,0 при x1=1,0, x2=2,0 и F5=14,0 при x1=1,0, x2 =3,0. Решения задач 4 и 5 удовлетворяют условиям целочисленности. Оптимальным является решение задачи 5, так как значение целевой функции F5=14,0>F4= 10,0. Для задачи 3 составляем новые задачи – задачи 6 и 7 (исключая значения 2<x2<3).

Задача 6

$F = 2x_1 + 4x_2 \to \max$

при ограничениях:

$$\begin{cases} 2x_1 + x_2 \le 19/3, \\ x_1 + 3x_2 \le 10, \end{cases}$$

$x_1 = 2,\ x_2 \le 2,$

x_1, x_2 – целые числа.

Задача 7

$F = 2x_1 + 4x_2 \to \max$

при ограничениях:

$$\begin{cases} 2x_1 + x_2 \le 19/3, \\ x_1 + 3x_2 \le 10, \end{cases}$$

$x_1 = 2,\ x_2 \ge 3,$

x_1, x_2 – целые числа.

Получили, что условия задачи 7 противоречивы, поэтому она исключается из дальнейшего рассмотрения. При решении задачи 6 получили целочисленные значения переменных x1=2,0; x2 =2,0 и целевая функции равна F6=12,0. Так как F6=12,0< F5=14,0, то решение задачи 6 не является оптимальным. Итак, прорешав все возможные варианты, определили, что оптимальным целочисленным решением является решение задачи 5 при x1=1,0; x2=3,0, оптимум функции F5=14,0.

Пример решения задачи целочисленного программирования в пакете Mathcad приведен на рисунках 1.4 и 1.5. На рисунке 1.5 – решение задачи целочисленного программирования с использованием встроенной функции floor(x), которая возвращает наибольшее целое число, меньшее или равное x.

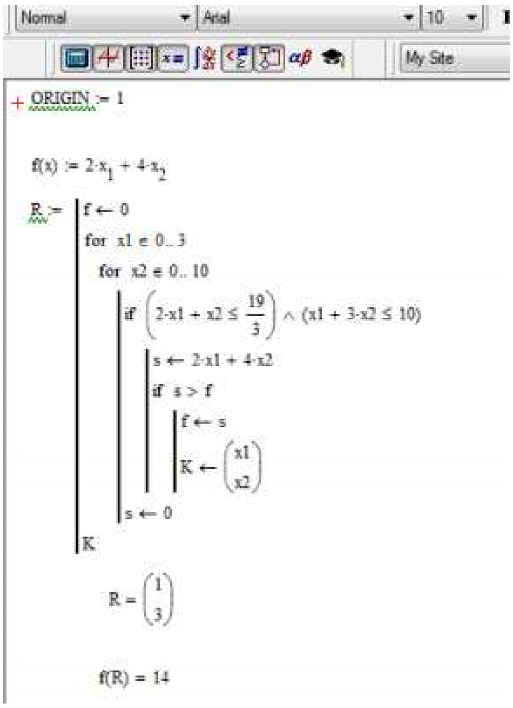


Рис. 1.4. Пример решения задачи целочисленного программирования в пакете Mathcad

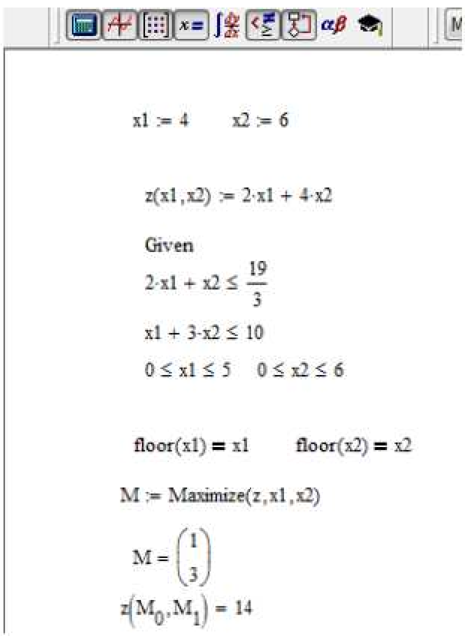


Рис. 1.5. Пример решения задачи целочисленного программирования в пакете Mathcad с использованием встроенной функции floor(x)

Приложение

Решить задачу линейного программирования с помощью MathCAD.

$$f(x) = 3x_1 + 2x$$

$$\begin{cases} x_1 - 2x_2 \le 1, \\ -x_1 + x_2 \ge -1 \\ x_1 + 3x_2 \le 8, \\ x_1 \ge 0, x_2 \ge 0 \end{cases}$$

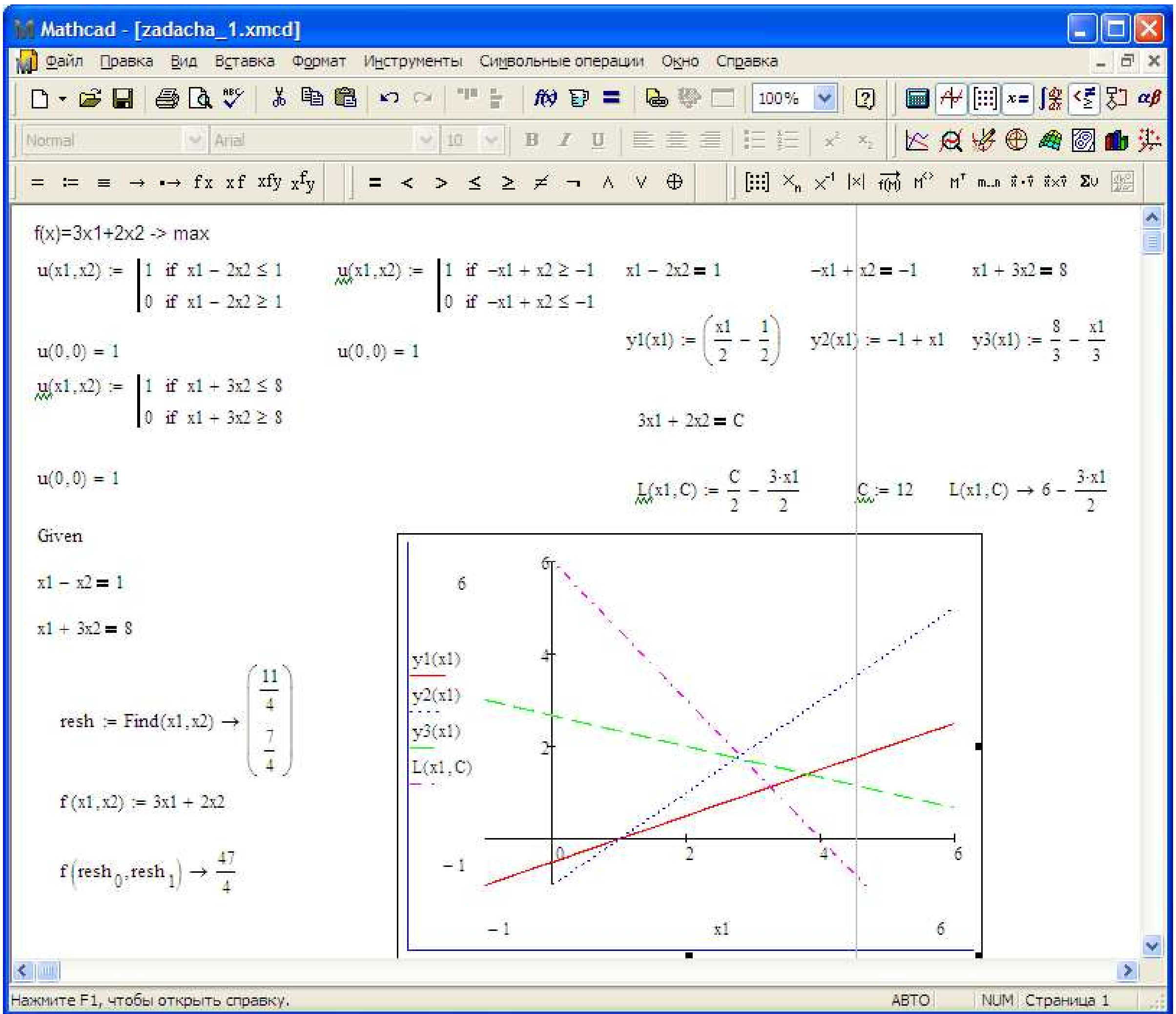


Рис 1.6. Решение задачи линейного программирования с помощью MathCAD

Контрольные задания

Задание 1

Решить методом Гомори-1 задачи целочисленного линейного программирования следующие задачи.

Во всех задачах, которые предлагаются ниже, все переменные неотъемлемые и целые.

1)
$$\begin{aligned} &5x_1 + 6x_2 + 6x_3 \to \min \\ &2x_1 + 4x_2 \geq 10 \\ &3x_1 + 2x_2 + 2x_3 \geq 10; \end{aligned}$$

2)
$$\begin{aligned} &6x_1 + 4x_2 \to \min \\ &2x_1 + x_2 \geq 3 \\ &x_1 - x_2 \leq 1; \end{aligned}$$

3)
$$\begin{aligned} &6x_1 + 4x_2 \to \min \\ &2x_1 + x_2 \geq 3 \\ &x_1 - 2x_2 \leq 2 \\ &3x_1 + 2x_2 \geq 1; \end{aligned}$$

4)
$$\begin{aligned} &x_1 + x_2 \to \min \\ &-x_1 - x_2 + x_5 = -1 \\ &-2x_1 + 2x_2 + x_3 = -2 \\ &-4x_1 + 2x_2 + x_4 = -1; \end{aligned}$$

5) $6x_1 + 4x_2 \to \min$

$2x_1 + x_2 \geq 3$

$x_1 - x_2 \geq 1;$

6) $2x_1 + 3x_2 \to \min$

$x_1 + 2x_2 \geq 16$

$2x_1 + x_2 \geq 16;$

7) $5x_1 - 3x_2 \to \max$

$3x_1 + 2x_2 \geq 6$

$2x_1 - 3x_2 \geq -6$

$x_1 - x_2 \leq 4$

8) $5x_2 + 7x_4 \to \min$

$-10x_2 + x_3 + x_4 = -16$

$x_1 - 3x_2 - 3x_4 = -12$

$-6x_2 - 2x_4 + x_5 = -17;$

9) $-5x_4 - 7x_5 \to \max$

$x_1 - x_4 - 2x_5 = -7$

$-x_3 + 3x_4 - 6x_5 = -3$

$x_2 - x_4 - 4x_5 = -11;$

10) $8x_1 + 2x_2 \to \max$

$x_1 - 4x_2 + x_3 = 4$

$-4x_1 + x_2 + x_4 = 4$

$x_1 + x_2 + x_5 = 6.$

<u>Задание 2</u>

Решить методом Гомори-2 задачи частично целочисленного линейного программирования следующие задачи.

1) $x_1 + 8x_2 \to \max$

$3x_1 + x_2 \leq 9$

$0.16x_1 + x_2 \leq 1.9$

$x_j \geq 0$, x_j — целое, $j = 1,2;$

2) $-6x_1 - x_2 \to \min$

$-2.9x_1 + 6x_2 \leq 17.4$

$3x_1 - x_2 \leq 1$

$x_j \geq 0$, x_j — целое, $j = 1,2;$

3) $0.25x_1 + x_2 \to \max$

$0.5x_1 + x_2 \leq 1.75$

$x_1 + 0.3x_2 \leq 1.5$

$x_j \geq 0$, x_j — целое, $j = 1,2;$

4) $-2x_1 - 4x_2 \to \min$

$2x_1 + x_2 \leq 19.33$

$x_1 + 3x_2 \leq 10$

$x_j \geq 0$, x_j — целое, $j = 1,2;$

5) $x_1 + x_2 \to \max$

6) $x_1 + x_2 \to \max$

$2x_1 + 11x_2 \le 38$

$x_1 + \quad x_2 \le 7$

$4x_1 - 5x_2 \le 5$

$x_j \ge 0, j = 1,2, x2$ — целое;

$2x_1 + 11x_2 \le 38$

$x_1 + \quad x_2 \le 7$

$4x_1 - 5x_2 \le 5$

$x_j \ge 0, j = 1,2, x_1$ — целое;

7) $x_1 \rightarrow \max$

$x_1 + 3x_2 \le 12$

$3x_1 - 8x_2 \le 24$

$x_j \ge 0, j = 1,2, x_1$ — целое;

8) $-8x_1 - 6x_2 \rightarrow \min$

$3x_1 + 5x_2 + x_3 = 11$

$4x_1 + x_2 + x_4 = 8$

$x_j \ge 0, j = 1,2, x1$ — целое.

<u>Задание 3</u>

Решить методом Гомори-3 задачи целочисленного линейного программирования следующие задачи.

Во всех задачах, которые предлагаются ниже, все переменные неотъемлемые и целые.

1) $-x_1 - 5x_2 \rightarrow \max$

$12x_1 - 3x_2 \ge 8$

$-3x_1 + 9x_2 \ge 8$

$-x_1 + 3x_2 \ge 3;$

2) $3x_1 + 2x_2 \rightarrow \min$

$-3x_2 \le 2$

$-2x_1 + 2x_2 \le 2$

$2x_1 - x_2 \ge 1;$

3) $x_1 + 3x_2 \rightarrow \min$

$x_1 + 6x_2 \ge 5$

$-5x_1 + 19x_2 \ge 13$

$-3x_1 + 6x_2 \ge 2;$

4) $7x_1 + 12x_2 \rightarrow \min$

$4x_1 - 23x_2 \le -14$

$-12x_1 + 2x_2 \le -9$

$-13x_1 - x_2 \le -10;$

5) $-3x_1 - 8x_2 \rightarrow \max$

$-2x_1 + 10x_2 \ge 4$

$-3x_1 - 5x_2 \le -10$

$2x_1 - x_2 \ge 2;$

6) $10x_1 + 7x_2 \rightarrow \min$

$2x_1 - 12x_2 \le -9$

$x_1 + 4x_2 \ge 10$

$4x_1 - 2x_2 \ge 14;$

7) $-3x_4 - 2x_5 \rightarrow \max$

$x_1 - 5x_4 + x_5 = -15$

8) $3x_2 + x_5 \rightarrow \min$

$-2x_2 + x_3 - 18x_5 = -20$

$x3 + x_4 - 8x_5 = -9$ $\qquad$ $x_1 + x_2 \quad - 15x_5 = -7$

$x_2 \quad - x_4 - 10x_5 = -19;$ $\qquad$ $-5x_2 \quad + x_4 + \quad x_5 = -9.$

2. Примеры прикладных задач с условиями целочисленности

2.1. Задача о рюкзаке

Представим себе путешественника, который собирает в дорогу рюкзак. В его распоряжении n предметов. Пусть известны вес a_j и «ценность» c_j каждого предмета (j=1, 2, …, n). Требуется заполнить рюкзак, не превышая его грузоподъемности b и максимизируя суммарную ценность груза. Получаем следующую булеву ЗЦЛП (задача о {0, 1}-рюкзаке):

$$\max(c_1x_1 + c_2x_2 + \ldots + c_nx_n)$$

$$\begin{cases} a_1x_1 + a_2x_2 + \ldots + a_nx_n \le b, \\ x_j \in \{0, 1\} \quad (j = 1, 2, \ldots, n). \end{cases}$$

Предположим теперь, что каждый предмет имеется в неограниченном числе экземпляров, и требуется также максимизировать суммарную ценность груза, не превышая грузоподъемности. Получаем так называемую *задачу о целочисленном рюкзаке:*

$$\max(c_1x_1 + c_2x_2 + \ldots + c_nx_n)$$

$$\begin{cases} a_1x_1 + a_2x_2 + \ldots + a_nx_n \le b, \\ x_j \in \mathbf{Z} \quad (j = 1, 2, \ldots, n). \end{cases}$$

Эта задача решается с помощью метода ветвей и границ.

2.2. Задача с фиксированными доплатами

В задаче линейного программирования функция представляет линейную форму:

$$f(x)=cx=c_1x_1+c_2x_2+\ldots+c_nx_n.$$

Однако, часто встречаются практические задачи, в которых целевая функция имеет вид

$$f(x) = \begin{cases} 0, & \text{если} \quad x = 0, \\ c_0 + cx, & \text{если} \quad x > 0, \end{cases}$$

где c_0 – некоторая фиксированная величина. Например, стоимость перевозки товара может состоять из двух частей: первая фиксирована и не зависит от объема перевозимого товара, а вторая – зависит. Предположим, что на переменные наложено требование неотрицательности $x \ge 0$ и известна верхняя оценка $x \le \hat{x}$ на величину неизвестных. Тогда путем введения новой целочисленной переменной такую задачу можно свести к ЗЦЛП.

Действительно, пусть переменная y удовлетворяет условиям:

$$y \in \{0, 1\},$$
$$y = 1 \quad \Leftrightarrow \quad x > 0.$$

Получаем

$$f(x) = c_0y + cx.$$

При расчете затрат на производство естественным считается учитывать фиксированные затраты или затраты на подготовку к осуществлению определенного вида деятельности. Такие затраты имеют место, как правило, когда продукция выпускается небольшими партиями и периодически возникает необходимость в переналадке оборудования. В данных случаях общие затраты – это сумма переменных затрат, связанных с определенным уровнем активности (расход сырья, материалов, затраты на оплату труда и т.д.), и постоянных расходов, необходимых для осуществления подобной деятельности.

Рассмотрим пример, иллюстрирующий данную ситуацию.

Компания «Сафари» по производству одежды может производить три типа одежды: рубашки, шорты и брюки. Для производства каждого вида одежды необходимо оборудование различных типов. Данное оборудование может быть арендовано по следующим ценам: для производства рубашек – 200 у.е. в неделю, для производства шортов – 150 у.е., для производства брюк – 100 у.е. Затраты материи и рабочей силына каждый из видов продукции приведены в таблице:

Продукция	**Труд (часы)**	**Материя (м^2)**
Рубашка	3	4
Шорты	2	3
Брюки	6	4

В распоряжении компании еженедельно имеется 150 часов рабочего времени и 160 м^2 материи. Переменные затраты и цена продажи на единицу продукции каждого вида приведены в таблице:

Продукция	***Цена продажи, у.е.***	***Переменные затраты, у.е.***
Рубашка	*12*	*6*
Шорты	*8*	*4*
Брюки	*15*	*8*

Сформулируйте модель целочисленного программирования для максимизации недельной прибыли компании «Сафари».

Решение.

Определим переменные

x1 – количество рубашек, производимых еженедельно;

x2 – количество шортов, производимых еженедельно;

x3 – количество пар брюк, производимых еженедельно.

Заметим, что стоимость аренды оборудования зависит только от типа производимой одежды, но не от ее количества. Это дает нам возможность использовать переменные

$$y_1 = \begin{cases} 1, & \text{если выпускается одна или больше рубашек}, \\ 0, & \text{если не выпускается}, \end{cases}$$
$$y_2 = \begin{cases} 1, & \text{если хотя бы одна единица шортов выпускается}, \\ 0, & \text{если не выпускается}, \end{cases}$$
$$y_3 = \begin{cases} 1, & \text{если выпускается одна пара брюк или больше}, \\ 0, & \text{если не выпускается}. \end{cases}$$

С помощью этих переменных можно отразить затраты на аренду оборудования. Переменные yi будут отличны от 0 в том случае, когда соответствующие xi также отличны от нуля. Недельная прибыль компании выразится с помощью следующей функции:

$$\max Z = 12x_1 + 8x_2 + 15x_3 - 6x_1 - 4x_2 - 8x_3 - 200y_1 - 150y_2 - 100y_3 =$$
$$= 6x_1 + 4x_2 + 7x_3 - 200y_1 - 150y_2 - 100y_3.$$

Два типа ограничений должны отражать условия по использованию в производстве рабочего времени (1) и материи (2):

3x1 + 2x2 + 6x3 $\leq$ 150 (труд),

4x1 + 3x2 + 4x3 $\leq$ 160 (материя).

В модели должна также найти отражение связь между производством конкретного типа продукции и арендой соответствующего оборудования. Для того чтобы учесть это условие, введем большие числа M1, M 2 и M 3.

Ограничения, показывающие вышеуказанную связь, запишутся следующим образом:

x1 $\leq$ M 1 y1 ,

$$x2 \le M\,2\;y2\;,$$

$$x3 \le M\,3\;y3\;.$$

Проверим, как «работают» данные ограничения. Так, если, например, $x1 > 0$, то переменная y1 не может быть равной 0.

Для случая, когда $y1 = 0$, переменная x1 должна быть меньше или равна 0:

$$x1 < 0 \text{ или } x1 = 0\;.$$

Таким образом, если $x1 > 0$, то $y1 = 1$. Если хотя бы одна рубашка производится, т.е. $x1 > 0$, то это будет означать, что $y1= 1$, и, следовательно, затраты на аренду будут включены в целевую функцию.

Заметим, что, если $y1= 1$, то это приводит к тому, что $x1 \le M1$. Это выражение, тем не менее, не должно ограничивать значение x1 сверх имеющихся ограничений, наложенных на данную переменную. Если M1 выбрано не очень большим, то такая ситуация может возникнуть, что не является верным. Поэтому значения Mi подбираются равными максимально допустимым значениям выпуска соответствующих видов продукции (исходя из мощности и наличия сырья).

В нашей задаче

$$M_1 = \min\{150/3; 160/4\} = \min\{50; 40\} = 40\,,$$

$$M_2 = \min\{150/2; 160/3\} = \min\{75; 53.3\} = 53 \text{ и}$$

$$M_3 = \min\{150/6; 160/4\} = \min\{25; 40\} = 25\,.$$

При $x1= 0$ ограничение $x1 \le M\,1\;y1$ превращается в $0 \le M1\;y1$. Это позволяет y1 принять значение либо 0, либо 1. Так как $y1 = 0$ менее «затратно» для оптимального решения, то нет сомнений, что при $x1 = 0$ переменная y1 будет равна 0. В окончательном виде модель задачи выглядит следующим образом:

$$\max Z = 6x_1 + 4x_2 + 7x_3 - 200y_1 - 150y_2 - 100y_3\,,$$

$$3x_1 + 2x_2 + 6x_3 \le 150,$$

$$4x_1 + 3x_2 + 4x_3 \le 160,$$

$$x_1 \le 40y_1\,,$$

$$x_2 \le 53y_2\,,$$

$x_3 \le 25y_3$,

$x_1, x_2, x_3 \ge 0$,

x_1, x_2, x_3 – целые,

y_1, y_2, y_3 – 0 или 1.

Оптимальное решение задачи Z = 75, x3 = 25, y3= 1. Таким образом, компания должна производить ежедневно по 25 пар брюк.

Рассмотренная задача относится к задачам с **фиксированными расходами**, когда затраты ассоциируются с выполнением некоторого действия на уровне, отличном от нуля, независимо от того, каков этот уровень активности.

В общем виде условие включения в целевую функцию или ограничение фиксированных затрат может быть представлено следующим образом:

$$f_j(x_j) = \begin{cases} c_j x_j + k_j, & \text{если } x_j > 0 \\ 0, & \text{если } x_j = 0, \end{cases}$$

где x_j определяет уровень активности j ($x_j \ge 0$); k_j – затраты, ассоциирующиеся с данной активностью (например, затраты на подготовительные операции, связанные с наладкой и переналадкой оборудования, затраты на аренду и т.д.).

Проблемы, в которых человек, принимающий решение, должен выбрать место, где расположить мощности, часто относятся к числу задач с фиксированными затратами, при этом затраты связываются со строительством или управлением данными мощностями.

2.3. Задача о «раскрое»

Из некоторого материала требуется изготовить m различных изделий в количестве, пропорциональном числам $b_1, \dots, b_m$. Каждая единица материала может быть раскроена n различными способами, причем использование j-го способа дает a_{ij} единиц i-го изделия. Найти план раскроя *a* единиц материала, обеспечивающий максимальное число комплектов изделий.

Обозначим черех x_j количество единиц материала, раскраиваемого j-ым способом. Переменные x_j должны удовлетворять следующим условиям:

$$\begin{cases} x_j \geq 0, \quad x_j \in \mathbf{Z}, \\ \sum\limits_{j=1}^{n} x_j = a, \\ \sum\limits_{j=1}^{n} a_{ij} x_j = b_i x \quad (i = 1, \ldots, m), \end{cases}$$

где x – число комплектов изделий.

Задача заключается в максимизации x при указанных ограничениях.

2.4. Задача коммивояжера

Постановка задачи. Пусть имеется n городов и известно расстояние между любой парой городов $c_{ij}, i, j = \overline{1, n}$. Расстояние между городами удобно записывать в виде матрицы n-го порядка C. Коммивояжер, выезжая из какого-либо города, должен посетить все n городов, побывав в каждом из них только один раз, и вернуться в исходный пункт, при этом длина маршрута должна быть *минимальной*.

Замечание 1. Если прямого сообщения между городами i и j нет, то полагают $c_{ij} = \infty$.

Замечание 2. Если городам поставить в соответствие вершины некоторого графа, а соединяющим их дорогам – дуги, то в терминах теории графов задача заключается в определении гамильтонова контура минимальной длины, в котором под длиной контура понимается не количество дуг, входящих в контур, а их суммарная длина.

Замечание 3. Так как решается задача минимизации длины маршрута, то естественно задать элементы матрицы C, стоящие на ее главной диагонали, равными ∞.

Формализуем задачу. Введем неизвестные:

$$x_{ij} = \begin{cases} 1, & \text{если коммивояжер переезжает из города } i \text{ в город } j, \\ 0, & \text{в противном случае.} \end{cases}$$

Тогда целевая функция Z примет вид

$$Z = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij}\, x_{ij} \to \min .$$

На матрицу неизвестных X накладываются ограничения:

$\sum_{i=1}^{n} x_{ij} = 1,\ j = \overline{1,n}$ (коммивояжер въезжает в каждый город только один раз);

$\sum_{j=1}^{n} x_{ij} = 1,\ i = \overline{1,n}$ (коммивояжер выезжает из каждого города только один раз);

$x_{ij} \in \{0, 1\},\ i, j = \overline{1,n}$.

Однако этих ограничений недостаточно, т.к. они не учитывают требование полного цикла, включающего все города. Устранение подциклов достигается добавлением ограничений

$$u_i - u_j + n\, x_{ij} \le n - 1,\ i, j = \overline{1,n},$$

где u_i могут принимать произвольные значения, в том числе и неотрицательные, и целые.

Задача коммивояжера в математической постановке эквивалентна задаче упорядочения конечного числа работ на машине с учетом потерь от переналадки. Длительность переналадки машины перед выполнением некоторой работы обычно зависит от характера предшествовавшей работы. Примером такой задачи о переналадке может быть задача о красках.

Пример. Пусть выпускаются четыре вида красок. Длительность очистки оборудования при перемене цвета представлена в таблице 2.1.

Таблица 2.1

Предшествовавший цвет	Последующий цвет			
	1. Белый	2. Желтый	3. Красный	4. Голубой
1. Белый	0	1	2	3
2. Желтый	6	0	1	2
3. Красный	8	6	0	1
4. Голубой	10	8	6	0

Требуется найти такую последовательность производства красок, при которой суммарные потери от переналадок были бы минимальными.

Метод ветвей и границ решения задачи коммивояжера

Существуют различные версии метода ветвей и границ решения задачи коммивояжера. Рассмотрим стандартный метод Дж. Литтла и др. (1963 г.).

Основная идея метода состоит в том, что вначале строят нижнюю границу длин маршрутов для всего множества гамильтоновых контуров *K*. Затем множество *K* разбивают на два подмножества таким образом, чтобы первое подмножество K_{ij}^{1} состояло из гамильтоновых контуров, содержащих некоторую дугу *(i,j)*, а другое подмножество $K_{\overline{ij}}^{1}$ не содержало этой дуги.

Для каждого из подмножеств определяют нижние границы по тому же правилу, что и для первоначального множества. Полученные нижние границы подмножеств K_{ij}^{1} и $K_{\overline{ij}}^{1}$ будут не меньше нижней границы для всего множества гамильтоновых контуров. Сравнивая нижние границы подмножеств K_{ij}^{1} и $K_{\overline{ij}}^{1}$, можно выделить среди них то, которое с большей вероятностью содержит оптимальный план («перспективное» подмножество).

Это «перспективное» подмножество по аналогичному правилу разбивается на два новые K_{ij}^{2} и $K_{\overline{ij}}^{2}$. Для них снова отыскивают нижние границы и т.д. Процесс ветвления продолжается до тех пор, пока не отыщется единственный полный гамильтонов контур. Получив его, просматривают отброшенные подмножества. Если их нижние границы больше длины полученного контура, то задача решена. Если есть такие, для которых нижние границы меньше этой длины, то найденное подмножество подвергается дальнейшему ветвлению, пока не убедимся, что оно не содержит лучшего гамильтонова контура. Если же такой найдется, то анализ оборванных ветвей продолжается относительно нового значения длины контура. Процесс решения будет закончен тогда, когда проанализируем все подмножества.

Применим описанный метод для решения примера о красках.

Матрица *C* имеет вид:

Таблица 2.2

j / i	1	2	3	4	u_i
1	∞	1	2	3	1
2	6	∞	1	2	1
3	8	6	∞	1	1
4	10	8	6	∞	6

Приведем ее по строкам:

Таблица 2.3

j / i	1	2	3	4
1	∞	0	1	2
2	5	∞	0	1
3	7	5	∞	0
4	4	2	0	∞
v_j	4	0	0	0

Приведем по столбцам:

Таблица 2.4

j / i	1	2	3	4
1	∞	0	1	2
2	1	∞	0	1
3	3	5	∞	0
4	0	2	0	∞

Вычислим константу приведения, которая будет нижней границей множества всехдопустимых гамильтоновых контуров: $s=1+1+1+6+4=13$.

Найдем степени нулей приведенной матрицы:

$$d_{12} = 1 + 2 = 3,\ d_{23} = 1 + 0 = 1,$$

$$d_{34} = 1 + 3 = 4,\ d_{41} = 1 + 0 = 1,\ d_{43} = 0 + 0 = 0.$$

Максимальный из них соответствует дуге *(3,4)*, которая и является претендентомна включение в гамильтонов контур.

Множество всех гамильтоновых контуров разделим на два: K_{34}^1 и $K_{\overline{34}}^1$. Матрицу, соответствующую K_{34}^1, с дугой *(3,4)* получаем из приведенной матрицы, вычеркивая строку 3 и столбец 4, а также меняя элемент *(4,3)* на ∞ (таблица 2.5).

Таблица 2.5

j / i	1	2	3
1	∞	0	1
2	1	∞	0
4	0	2	∞

Матрицу, соответствующую $K_{\overline{34}}^1$, получим из таблицы 2.4 путем замены элемента c_{34} на ∞ (таблица 2.6).

Таблица 2.6

j / i	1	2	3	4
1	∞	0	1	2
2	1	∞	0	1
3	3	5	∞	∞
4	0	2	0	∞

Матрица таблицы 2.6 приведена, поэтому нижняя граница множества K_{34}^1

$$s\left(K_{34}^1\right) = 13 + 0 = 13.$$

Находим константу приведения для множества контуров $K_{\overline{34}}^1$ по таблице 2.6:

$$s\left(K_{\overline{34}}^1\right) = 13 + 3 + 1 = 17.$$

Сравниваем нижние границы подмножеств K_{34}^1 и $K_{\overline{34}}^1$. Так как s(K_{34}^1)<s($K_{\overline{34}}^1$), то дальнейшему ветвлению подвергаем множество K_{34}^1. Найдем степени нулей в таблице 2.5:

$$d_{12} = 1 + 2 = 3,\; d_{23} = 1 + 1 = 2,\; d_{41} = 2 + 1 = 3.$$

Максимальную степень имеют два нуля. Выберем дугу *(1,2)* и множество K_{34}^{1} разобьем на два: K_{12}^{2} и $K_{\overline{12}}^{2}$. Первому подмножеству соответствует таблица 2.7, второму – таблица 2.8.

Таблица 2.7

j / i	1	3
2	∞	0
4	0	∞

Константа приведения для этой таблицы не меняется: s(K_{12}^{2})=13+0=13.

Таблица 2.8

j / i	1	2	3
1	∞	∞	1
2	1	∞	0
4	0	2	∞

Здесь s($K_{\overline{12}}^{2}$)=13+2+1=16.

Так как s(K_{12}^{2})<s($K_{\overline{12}}^{2}$), то перспективной является ветвь, соответствующая множеству K_{12}^{2}. Однако матрица таблицы 2.7 является приведенной и содержит два нулевых элемента.

По таблице 2.7 можно заключить, что в оптимальный гамильтонов контур должны войти дуги *(2,3)* и *(4,1)*.

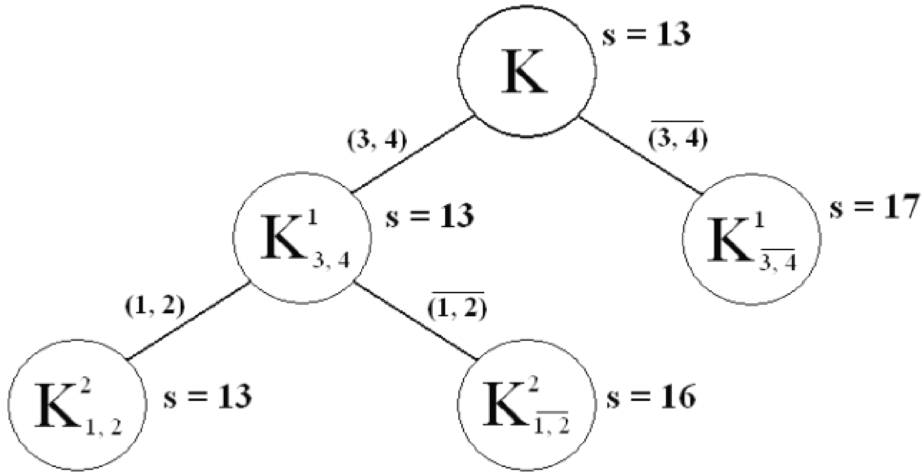
Таким образом, в оптимальный гамильтонов контур входят дуги *(3,4)*, *(1,2)*, *(2,3)*, *(4,1)*. После упорядочения этих дуг, начиная, например, с дуги *(1,2)*, получим цикл

(1,2), (2,3), (3,4), (4,1),

т. е. последовательность изготавливаемых красок должна быть следующей: белая, желтая, красная и голубая. При этом затраты на переналадку составят 13 ед.

Замечание. Цикл можно начинать с любой дуги оптимального контура.

Ветвление, использованное при решении данной задачи, можно представить в следующем виде:



2.5. Задача перекрытия набора элементов

В проблемах такого типа каждый элемент из заданного множества элементов должен быть «перекрыт» приемлемым элементом некоторого другого множества. Цель состоит в том, чтобы определить минимальное число перекрывающих элементов. Рассмотрим пример.

В некотором районе имеется 6 городов. Необходимо определить, где построить пожарные станции. В районе должно быть построено минимальное количество этих станций. При этом необходимо учитывать условие, чтобы, по крайней мере, одна пожарная станция находилась в 15 минутах езды от каждого из городов. Время езды между городами района приведено в таблице.

Таблица 2.9

Из города	*В город*					
	1	*2*	*3*	*4*	*5*	*6*
1	*0*	*10*	*20*	*30*	*30*	*20*
2	*10*	*0*	*25*	*35*	*20*	*10*
3	*20*	*25*	*0*	*15*	*30*	*20*
4	*30*	*35*	*15*	*0*	*15*	*25*
5	*30*	*20*	*30*	*15*	*0*	*14*
6	*20*	*10*	*20*	*25*	*14*	*0*

Требуется сформулировать модель целочисленного программирования для определения числа пожарных станций, которые следует построить в районе, и мест расположения этих станций.

Решение.

Для каждого города определим переменную xi , принимающую значение 0 или 1 и определяющую строительство (xi = 1) или нестроительство (xi = 0) пожарной станции в i-м городе.

Общее количество пожарных станций определяется суммой

$$Z = x1 + x2 + x3 + x4 + x5 + x6 ,$$

оно должно быть минимальным, т.е.

$$\min Z = x1 + x2 + x3 + x4 + x5 + x6 .$$

Ограничения модели должны отражать условие того, чтобы пожарная станция находилась в 15 минутах езды от каждого города. Для построения ограничений рассмотрим таблицу, в которой отразим достижимость городов из каждого другого города в 15-минутный интервал времени:

Город отправления	Город назначения
1	1,2
2	1,2,6
3	3,4
4	3,4,5
5	4,5,6,
6	2,5,6

С учетом данных этой таблицы построим ограничения:

Для города 1 $x_1 + x_2 \geq 1$ (хотя бы одна станция находится в 15 минутах езды от городов 1 и 2) для города 2 $x_1 + x_2 + x_6 \geq 1,$

для города 3 $x_3 + x_4 \geq 1,$

для города 4 $x_3 + x_4 + x_5 \geq 1,$

для города 5 $x_4 + x_5 + x_6 \geq 1,$

для города 6 $x_2 + x_5 + x_6 \geq 1,$

$$x_i = 0 \vee 1, \ i = 1, \ldots, 6 .$$

Оптимальное решение задачи $Z = 2$, $x_2 = x_4 = 1$, $x_1 = x_3 = x_5 = x_6 = 0$.

Таким образом, необходимо построить только две пожарные станции во втором и четвертом городах.

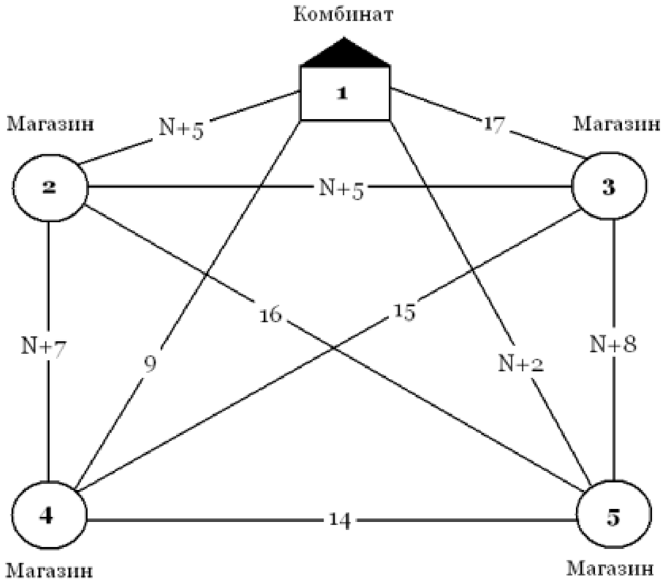
Контрольные задания

Вариант 1. Продавец книг, проживающий в городе A, один раз в месяц должен посетить своих четырех клиентов, которые проживают в городах B, C, D, E соответственно. Приведенная ниже таблица содержит расстояния в километрах между различными городами (N – порядковый номер студента в группе).

	A	B	C	D	E
A	–	N+12	22	N+15	21
B	N+12	–	N+8	11	N+13
C	22	N+8	–	16	N+18
D	N+15	11	16	–	19
E	21	N+13	N+18	19	–

Необходимо составить такой маршрут движения продавца книг, чтобы суммарное пройденное им расстояние было минимальным.

Вариант 2. Молочный комбинат осуществляет доставку продукции в ряд торговых точек города автомобильным транспортом. Определите маршрут минимальной длины доставки продукции во все торговые точки и возвращения транспортного средства на комбинат для очередной загрузки. Расстояния между торговыми точками и комбинатом (в километрах) известны и представлены следующим графом **(**N – порядковый номер студента в группе**).**



Библиографический список.

1. Костюкова О.И. Исследование операций: Учеб. пособие для студ. спец. 31 03 04 «Информатика» всех форм обучения. – Мн.: БГУИР, 2003, 94 с.: ил.

2. Богданова Е.Л. Оптимизация в проектном менеджменте: линейное программирование: учебное пособие / Е.Л. Богданова, К.А. Соловейчик, К.Г. Аркина. – СПб.: Университет ИТМО, 2017. – 165 с.

3. Шевченко В.Н., Золотых Н.Ю. Линейное и целочисленное линейное программирование. — Нижний Новгород: Изд-во Нижегородского госуниверситета им. Н.И. Лобачевского, 2004. — 154 с..

4. Ефимова Г. А., Страхов Е. М. Целочисленное программирование: методические указания и варианты заданий для самостоятельной работы студентов направлений подготовки 6.040201 «математика», 6.040301 «прикладная математика». Одесский национальный университет имени И. И. Мечникова, 2014. - 40 с.

5. Струченков В.И. Методы оптимизации в прикладных задачах. М.: СОЛОН┐ ПРЕСС, 2009. – 320 с.

6. Параев Ю.И., Панасенко Е.А. Методы оптимизации – Томск: Изд-во ТУСУР, 2012, – с. 20.

Электронное учебное издание

Ольга Викторовна **Свиридова**
Александр Александрович **Рыбанов**

Целочисленное программирование

Учебное пособие

Электронное издание сетевого распространения

Редактор Матвеева Н.И.

Темплан 2022 г. Поз. № 2.
Подписано к использованию 13.07.2022. Формат 60x84 1/16.
Гарнитура Times. Усл. печ. л. 4,0.

Волгоградский государственный технический университет.
400005, г. Волгоград, пр. Ленина, 28, корп. 1.

ВПИ (филиал) ВолгГТУ.
404121, г. Волжский, ул. Энгельса, 42а.