### Глава 1. Задачи линейного программирования

* 1. Постановка задачи линейного программирования.

Примеры.

*Линейное программирование* - это направление математического программирования, изучающее методы решения экстремальных задач, которые характеризуются линейной зависимостью между переменными и линейным критерием.

В общей постановке задачи линейного программирования формулируется следующим образом.

Имеются какие-то переменные =(*x*1, *x*2,…, *xn*) и линейная функция этих переменных, которая носит название целевой функции. Ставится задача: найти экстремум (максимум или минимум) целевой функции при условии, что переменные удовлетворяют системе линейных равенств и/или неравенств.

Классическими примерами практических задач, сводящихся к задаче линейного программирования, являются задача о диете, а также задача о составлении плана производства.

В задаче о диете составляется наиболее экономный (т.е. наиболее дешевый) рацион питания животных, удовлетворяющий определенным медицинским требованиям. При этом в качестве переменных *x*1, *x*2,…, *xn*  выступают количества продуктов питания, используемых в рационе.

Задачу о составлении плана производства рассмотрим более подробно.

Пусть некоторая производственная единица (предприятие, цех, отдел и т.д.) может производить *n* видов товаров *G*1, *G*2,…, *Gn*, используя при этом *m* видов сырьевых ресурсов *R*1, *R*2,…,*Rm*, запасы которых ограничены величинами *b*1, *b*2,…,*bm*.

Tехнологией производства товара *Gj*назовем набор чисел *aij*, показывающий, какое количество *i*-го ресурса необходимо для производства единицы товара *Gj*. Это можно записать в виде технологической матрицы, которая полностью описывает технологические потребности производства и элементами которой являются числа *aij*.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | *G*1 | *G*2 | … | *G*тn |
| *R*1 | *a*11 | *a*12 | … | *a*11 |
| *R*2 | *a*21 | *a*22 | … | *a*2*n* |
| … | … | … | … | … |
| *R*m | *am*1 | *am*2 | … | *anm* |

Предположим также, что известны цены реализации единицы каждого товара *с*1, *с*2, …, *сn*.

Обозначим через *x*1, *x*2,…, *xn*  планируемое производство единиц товаров *G*1, *G*2,…, *Gn*. В силу имеющейся технологической матрицы для этого потребуется:

*a*11*x*1+*a*12*x*2+…+*a*1*nxn* – ресурса *R*1,

*a*21*x*1+*a*22*x*2+…+*a*2*nxn* – ресурса *R*2,

……………………………………….

*a*m1*x*1+*a*m2*x*2+…+*a*m*nxn* – ресурса *Rm*.

С учетом ограничений на запасы ресурсов, а также очевидных условий неотрицательности переменных *x*1, *x*2,…, *xn*  получим следующую систему линейных неравенств:



Естественно предположить, что целью производственной единицы является получение максимальной выручки за произведенную продукцию, т.е. максимизация функции:

*F*=*c*1*x*1+*c*2*x*2+…+*cnxn*.

Таким образом, с учетом естественного требования неотрицательности переменных получаем линейную оптимизационную задачу, которая может быть представлена в следующей формальной записи:

*F*=*c*1*x*1+*c*2*x*2+…+*cnxn*→max



Мы не будем рассматривать примеры других задач линейного программирования. Отметим лишь, что они встречаются очень часто при оптимизации самых разнообразных производственных и экономических задач.

* 1. Формы задач линейного программирования. Основные понятия и утверждения.

Задача линейного программирования математически может быть представлена в различных формах.

*Общей задачей ЛП* называется задача, которая состоит в определении максимального (минимального) значения функции

 (1.1)

при условиях

 (1.2)

 (1.3)

где *aij, bi, cj -* заданные постоянные величины и *k m*.

Функция (1.1) называется *целевой функцией* задачи (1.1)-(1.3), а условия (1.2)-(1.3) - *ограничениями* данной задачи.

Таким образом, ограничения в общей задаче ЛП заданы в виде линейных неравенств и/или линейных уравнений. При этом заметим, что в неравенствах, вообще говоря, могут применяться оба знака, как "≤", так и "≥". Однако домножением при необходимости обеих частей какого-либо неравенства на (-1) можно свести их все к единому виду (1.2).

Вектор *=(x1 , x2, ..., xn),* компоненты которого удовлетворяют ограничениям (1.2)-(1.3), называется *допустимым планом (или допустимым решением).*

Совокупность всех допустимых планов задачи линейного программирования образует *допустимое множество* решений этой задачи. Будем обозначать его через Х.

Допустимый план *=(x1\*, x2\*, ...xn\*),* при котором целевая функция задачи (1) принимает свое максимальное (минимальное) значение, называется *оптимальным.*

Решить задачу линейного программирования - это значит, найти все ее оптимальные планы или доказать их отсутствие.

Помимо общей формы различают еще две частные задачи линейного программирования - стандартную и основную.

Особенностью *стандартной задачи ЛП* является то, что ее ограничения представлены в виде линейных неравенств, а также условий неотрицательности на переменные, присутствующие в задаче:

 (1.4)

Ограничения *основной*  *задачи ЛП* представляют собой линейные ограничения-равенства, а также условия неотрицательности на переменные:

 (1.5)

Между различными формами задач линейного программирования существует тесная взаимосвязь, в смысле возможности перехода от одной формы записи к другой.

Например, сведение задачи минимизации функции к задаче максимизации осуществляется простым домножением целевой функции на (-1). То есть, если в задаче линейного программирования функция

→min,

то, введя в рассмотрение функцию , получим:

→max.

Переход от стандартной задачи к основной связан с введением дополнительных переменных *xn+i , i=*1*,*…,*m*. Покажем это на примере.

*Пример 1.1.* Исходная задача ЛП имеет вид:



В первом из неравенств левая часть не больше правой части. Поэтому добавлением в левую часть некоторого неотрицательного слагаемого *x*4 можно свести это неравенство к равенству. Аналогично во втором неравенстве левая часть не меньше правой. Следовательно, вычитая из левой части некоторое неотрицательное число *x*5, можно также свести данное неравенство к равенству. Итак, получим новую форму записи задачи ЛП (основную):



Основные утверждения теории линейного программирования касаются, в первую очередь, вида допустимого множества решений Х, а также существования и свойств оптимальных решений задачи. Изложим эти утверждения в интерпретации, близкой к наглядной.

Возможные случаи допустимого множества решений

задачи линейного программирования

1. Допустимое множество решений пусто.

Данному случаю соответствует взаимная противоречивость ограничений, входящих в задачу.

2) Допустимое множество - выпуклый ограниченный многогранник.

1. Допустимое множество - выпуклое неограниченное многогранное множество.

Два последних случая достаточно легко представить в двух- или трехмерном измерении. В пространстве большей размерности понятие многогранника (многогранного множества) вводится абстрактно как пересечение гиперплоскостей и гиперполуплоскостей, определяемых соответствующими линейными уравнениями и неравенствами, входящими в состав ограничений задачи. Характерным свойством многогранника является наличие в нем особых точек - *вершин*.

Возможные случаи оптимальных решений (планов) задачи

линейного программирования.

1) Задача *не имеет оптимальных решений*.

Данный случай может возникнуть: либо тогда, когда допустимое множество решений пусто ("не из чего выбирать" оптимальный план),

либо когда допустимое множество представляет собой неограниченное многогранное множество, и целевая функция на нем неограниченно возрастает (если *L*→ max) или неограниченно убывает (при *L*→min).

2) Задача имеет *единственное* *решение* (единственный оптимальный план).

Это решение обязательно совпадает с одной из вершин допустимого множества.

3) Задача имеет *бесконечное множество* оптимальных решений, заданное некоторым линейным образованием - ребром, гранью, гипергранью и т.д. Среди точек этого линейного образования имеются и вершины допустимого множества.

Таким образом, основное утверждение теории линейного программирования, в конечном итоге определяющее специфические способы его решения, можно сформулировать следующим образом:

Если задача линейного программирования имеет хотя бы один оптимальный план, то его следует искать среди вершин допустимого множества решений.

В следующем параграфе рассмотренные общие положения будут проиллюстрированы на примере задачи линейного программирования с двумя переменными.

* 1. Графоаналитический способ решения задач линейного программирования

Графоаналитический (графический) способ решения задач линейного программирования обычно используется для решения задач с двумя переменными, когда ограничения выражены неравенствами, а также задач, которые могут быть сведены к таким задачам.

Пусть задача линейного программирования имеет вид:

 (1.6)

 (1.7)

где *с*1, *с*2, *а*i1, *а*i2, *b*i - заданные действительные числа; знаки в неравенствах произвольны; целевая функция либо максимизируется, либо минимизируется.

Каждое из неравенств (1.7) системы ограничений задачи геометрически определяет полуплоскость соответственно с граничными прямыми ; *i*=1,…,*m*. В том случае, если система неравенств (1.7) совместна, допустимая область решений задачи есть множество точек, принадлежащих всем указанным полуплоскостям. Так как множество точек пересечения данных полуплоскостей – выпуклое, то областью допустимых значений является выпуклое множество, которое называют *многоугольником* решений. Стороны этого многоугольника лежат на прямых, уравнения которых получаются из исходной системы ограничений заменой знаков неравенств на знаки равенств.

Множеством допустимых решений для данной частной задачи может быть:

* пустая область;
* выпуклый многоугольник, включая вырожденные случаи - отрезок и единственную точку;
* выпуклая многоугольная неограниченная область, включая вырожденные случаи - луч и прямую.

Практическая реализация решения задачи линейного программирования (1.6) – (1.7) на основе ее геометрической интерпретации включает следующие этапы:

1.     Построить прямые, уравнения которых получаются в результате замены в ограничениях (1.7) знаков неравенств на знаки равенств.

2.     Найти полуплоскости, определяемые каждым из ограничений.

Соответствующая полуплоскость может быть найдена подстановкой в неравенство координат какой-нибудь «простой» точки - (0,0), (0,1) или (1,0). Главное - чтобы эта точка не принадлежала границе полуплоскости. Если после подстановки неравенство окажется справедливым, то выбирается та полуплоскость, где содержится эта точка. Если неравенство не справедлива, то выбирается альтернативная полуплоскость.

3.     Определить многоугольник решений, как пересечение найденных полуплоскостей.

4.     Построить градиент целевой функции, т.е. вектор , координатами которого служат коэффициенты целевой функции *L*.

Этот вектор определяет направление наискорейшего возрастания целевой функции.

5.     Построить ряд линий уровня целевой функции *L*, т.е. прямых перпендикулярных градиенту *L*. При этом построение линий уровня следует вести в направлении градиента, если решается задача на максимум, и в противоположном направлении (в направлении «антиградиента»), если решается задача на минимум. В результате отмечается точка (точки), в которой линии уровня в последний раз касаются допустимого множества.

Если допустимое множество неограниченно, то точки последнего касания может и не быть. Линии уровня уходят в бесконечность, соответственно значение  или , и задача не имеет оптимальных планов.

1. Определить координаты отмеченной точки аналитически, решая соответствующую систему линейных уравнений. Затем вычислить значение целевой функции в этой точке. Тем самым, определяется оптимальный план и оптимальное значение целевой функции задачи.

Заканчивая рассмотрение геометрической интерпретации задачи (1.6) – (1.7), отметим, что при нахождении ее решения могут встретиться случаи, изображенные на рис. 1.1 – 1.3. Рис. 1.1 характеризует такой случай, когда целевая функция принимает оптимальное значение в единственной точке А, одной из вершин допустимого множества. На рис. 1.2 оптимальное значение целевая функция принимает в любой точке отрезка АВ. На рис. 1.3 изображен случай, когда оптимальное значение целевой функции недостижимо.

А

*х*1

*х*2

# Рис. 1.1. Оптимум функции *L* достижим в точке А

А

*х*1

*х*2

В

Рис. 1.2. Оптимум функции *L* достигается в любой точке отрезка АB

*х*1

*х*2

Рис. 1.3. Оптимум функции *L* недостижим

* 1. Симплекс-метод

Симплекс-метод является универсальным методом решения задач линейного программирования с любым числом переменных и с любым числом ограничений.

Тем не менее, исходная форма задачи, к которой непосредственно применим симплекс-метод, должна иметь специальный вид. Эта форма является частным случаем основной формы (1.5) задач линейного программирования. Здесь также система ограничений представлена ограничениями-равенствами (линейными уравнениями) и условиями неотрицательности. Однако в равенствах, кроме того, выделяются так называемые *базисные переменные*. В каждом из равенств присутствует одна определенная базисная переменная, взятая с единичным коэффициентом, а в других равенствах ее нет. Число базисных переменных, таким образом, совпадает с числом ограничений-равенств в системе и обычно строго меньше общего числа переменных. Остальные переменные называются небазисными или свободными. Еще одно требование заключается в выполнении условия неотрицательности свободных членов *bi* в равенствах. Наконец, целевая функция задачи должна быть выражена только через небазисные переменные. Некоторые авторы называют такую форму представления задачи линейного программирования *канонической*.

Отметим, что во многих случаях каноническая форма задачи получается автоматически при переходе от стандартной формы (1.4) к основной с помощью введения новых переменных. Для этого требуется, чтобы свободные члены в неравенствах были неотрицательными, и все неравенства в (1.4) имели единственный знак «». Отметим, что модель задачи о составлении производственного плана, рассмотренная в параграфе 1.1 вполне соответствует этим требованиям.

*Пример 1.2.* Рассмотрим задачу линейного программирования вида:

*L*=3*x*1+4*x*2+6*x*3→max



Преобразуем первое и второе неравенства в равенства, введя новые неотрицательные переменные *x*4  и *x*5. Получим новую систему ограничений:



Переменная *x*3 входит с единичным коэффициентом только в первое уравнение, переменная *x*4 – только во второе уравнение. Именно они и составляют базис задачи. Целевая функция выражена лишь через небазисные переменные *x*1, *x*2, *x*3. Таким образом, имеем каноническую форму представления.

Решение зада­чи при помощи симплекс-метода распадается на ряд шагов. На каждом шаге от данного базиса переходят к другому, новому ба­зису  с таким расчетом, чтобы значение функции *L* улучшалось, т. е.увеличивалось (по крайней мере, не уменьшалось), если , и уменьшалось (не увеличивалось), если . Для перехода к новому базису из старого базиса уда­ляется одна из переменных и вместо нее вводится другая из числа небазисных. После конечного числа шагов находится некоторый ба­зис, на котором достигается искомый максимум (минимум) для линейной функции *L*, а соответствующее базисное решение является опти­мальным либо выясняется, что задача не имеет решения.

С геометрической точки зрения каждому каноническому представлению задачи линейного программирования (каждому базису) соответствует определенная вершина допустимого множества (многогранника решений), которая, согласно утверждениям параграфа 1.2, является «претендентом» на возможность быть оптимальным планом. Таким образом, решение задачи симплекс-методом заключается в последовательном и целенаправленном переходе от одной вершины допустимого множества к другой.

Абстрактным аналогом понятия вершины допустимого множества является понятие *опорного плана*.

Если задача линейного программирования представлена в канонической форме, то опорный план *=(x*1, *x*2,*..., xn),*, может быть получен с помощью простого правила: его базисные компоненты равны свободным членам ограничений-равенств, его небазисные компоненты равны нулю.

Итак, предположим, что исходная задача линейного программирования задана в канонической форме. Практическая реализация симплекс-метода обычно сводится к последовательному построению так *называемых симплекс-таблиц*, в которых отражается очередная каноническая форма представления исходной задачи, а также содержится проверка условия оптимальности соответствующего опорного плана. В многочисленных учебных пособиях по линейному программированию представлены весьма различные варианты оформления этих таблиц. Ниже дается один из таких вариантов.

Таблица 1.1.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Базис* | *x*1 | *…* | *xn* | *xn+*1 | *…* | *xn+m* | *Свободные члены* |
| *xn+*1 | *a*11 | *…* | *a*1n | 1 | *…* | 0 | *b*1 |
| *…* | *…* | *…* | *…* | *…* | *…* | *…* | *…* |
| *xn+m* | *am*1 | *…* | *amn* | 0 | *…* | 1 | *bm* |
|  | *-c*1 | *…* | *-cn* | 0 | *…* | 0 |

В первой строке обозначены названия столбцов. В следующих *m* строках отображаются коэффициенты перед переменными в ограниченях-равенствах задачи, а также свободные члены. В последней строке содержатся коэффициенты целевой функции, если решается задача на минимум, и коэффициенты целевой функции, взятые с противоположными знаками, если решается задача на максимум. В первом столбце перечисляются базисные переменные в соответствии с присутствием их в том или ином равенстве. Если симплекс-таблица составлена правильно, то в ней имеется *m* единичных столбцов (один элемент - единица, остальные элементы - нули), а свободные члены в ней неотрицательны.

Алгоритм симплекс-метода рассмотрим на решении задачи из примера 1.2. Соответствующая симплекс-таблица имеет вид:

Таблица 1.2.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Базис* | *x*1 | *х*2 | *х*3 | *x*4 | *х*5 | *Свободные члены* | *Симплекс-отношения* |
| *х*4 | 2 | 5 | 2 | 1 | 0 | 12 | 6\* |
| *х*5 | 7 | 1 | 2 | 0 | 1 | 18 | 9 |
|  | -3 | -4 | -6\* | 0 | 0 |  | |

По сравнению с таблицей 1.1 здесь добавлен еще столбец для симплекс-отношений, о чем речь пойдет ниже. В последней строке записаны коэффициенты целевой функции, взятые с противоположными знаками, так как .

Данной симплекс-таблице, согласно выше приведенному правилу, соответствует опорный план (вершина) вида

.

Условие оптимальности состоит в следующем: если в последней строке симплекс-таблицы все элементы неотрицательны, то соответствующий опорный план является оптимальным, и задача решена. В нашем случае условие оптимальности не выполняется, так как в последней строке имеется три отрицательных элемента. Поэтому необходимо перейти к новому опорному плану и соответственно построить новую симплекс-таблицу.

*Алгоритм симплекс-метода*

1.     В последней строке исходной симплекс-таблицы выбираем наименьший отрицательный элемент. Он отмечен знаком \*. Столбец, со­ответствующий этому элементу, называется *ведущим*. Он определяет переменную, которая будет введена в базис на данном этапе. Это - переменная *х*3.

2.     Вычисляют отношения свободных членов к элементам ведущего столбца (симплекс-отношение): *θ*1=12/2=6, *θ*1=18/2=9. Находят наименьшее *неотрицательное* из этих симплекс-отношений. Оно соответствует *ведущей* строке, которая определяет переменную, выводимую из базиса. Это - переменная *х*4.

3. Если все симплекс-отношения окажутся отрицательными, то задача не имеет решений (оптимум целевой функции не достигается).

4.     На пересечении ведущей строки и ведущего столбца находится ведущий элемент.

5.     Если имеется несколько одинаковых по величине симплекс-отношений, то выбирают любое из них. То же самое относится к отрицательным элементам последней строки симплекс-таблицы.

1. После нахождения ведущего элемента переходят к следующей таблице (табл. 1.3). Для этого вначале заполняем первый столбец, записывая новые базисные элементы: *х*3 и *х*5.
2. Далее, элементы ведущей строки табл. 1.2, за исключением симплекс-отношения, делим на ведущий элемент.
3. Остальные элементы ведущего столбца делаем равными нулю.

8.     Оставшиеся элементы симплекс-таблицы вычисляются по правилу прямоугольника: мысленно вычерчиваем прямоугольник в табл. 1.2, одна вершина которого совпадает с разрешающим элементом, а другая - с элементом, образ которого мы ищем; остальные две вершины определяются однозначно. Тогда искомый элемент из табл. 1.3 будет равен соответствующему элементу табл. 1.2 минус дробь, в знаменателе которой стоит разрешающий элемент, а в числителе - произведение элементов из двух неиспользованных вершин прямоугольника.

Таблица 1.3.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Базис* | *x*1 | *х*2 | *х*3 | *x*4 | *х*5 | *Свободные члены* | *Симплекс-отношения* |
| *х*3 | 1 | 2,5 | 1 | 0,5 | 0 | 6 |  |
| *х*5 | 5 | -4 | 0 | -1 | 1 | 6 |  |
|  | 3 | 11 | 0 | 3 | 0 |  | |

Например, элемент, стоящий на пересечении строки *х*3 и столбца *x*1 находим так: 7-2⋅2/2=5; свободный член в строке *х*5: 18-2⋅12/2=6; первый элемент последней строки: -3-2⋅(-6)/2=3 и т.д.

9. Записываем соответствующий опорный план:



и снова проверяем условие оптимальности. На этот раз условие выполнено: в последней строке все элементы неотрицательны. Значит найденный опорный план оптимален. Чтобы получить оптимальный план исходной, стандартной задачи, нужно отбросить последние два элемента из : .

10. Вычисляем оптимальное значение целевой функции: *L*\*=3⋅0+4⋅0+6⋅6=36.

### 1.5. Метод искусственного базиса (М-метод)

М-метод применяется для решения любых задач линейного программирования, в том числе и тех, где начальная каноническая форма не задана.

М-метод состоит во введении новых искусственных переменных, которые сразу можно взять в качестве базисных, и дальнейшем решении полученной задачи симплекс-методом.

Предположим, что исходная задача линейного программирования представлена в основной форме:



*Алгоритм М-метода:*

В каждое *i-*ое ограничение вводим искусственную переменную *хn+i* >0. Всего *m* новых искусственных переменных.

Если , то в целевую функцию *L* вводим *m* дополнительных слагаемых вида: -M⋅*xn*+1, -M⋅*хn*+2, ..., -M⋅*хn+m*; если же , то слагаемые вида: M⋅*xn*+1, M⋅*хn*+2, ..., M⋅*хn+m*, где М - произвольная очень большая константа.

Получим новую, вспомогательную задачу линейного программирования:

  
  
F(X) = c1Х1 + ... + сnXn -M\*Xn+1 - ... -M\*Xn+m => max   
  
ai,1X1+ ... + ai,nXn +Xn+i = bi , (i=1,m)   
  
Xj >0 , (j=1,n+m)   
  
Новая система ограничений характерна тем, что искусственные переменные сразу можно взять в качестве базисных:

Формируем начальное базисное решение новой М-задачи :   
  
X' = ( 0, ... 0, b1, ... bm )

Решаем М-задачу [симплекс-методом](http://www.tisbi.ru/resources/lib/linprog/theory2.htm)

Анализируем решение М-задачи в соответствии со следующими правилами :

Если в оптимальном решении М-задачи:   
  
X" = ( X"1, ... X"n, X"n+1, ... X"n+m )   
  
все искусственные переменные равны 0, то вектор   
  
X" = ( X"1, ... X"n )   
  
является оптимальным решением исходной ЗЛП.

Если в оптимальном решении М-задачи хотя бы одна искусственная переменная не равна 0, то исходная ЗЛП не имеет решения в силу несовместимости ограничений.

Если М-задача не имеет решения , то исходная ЗЛП также не имеет решения в силу неограниченности целевой функции на допустимом множестве.

Исходная [ЗЛП](http://www.tisbi.ru/resources/lib/linprog/theory2.htm#ZLP):   
  
F(X) = 4X1 +4X2 +2X3 +4X4 +3X5 +1X6 => max   
  
-3X1 +0X2 +2X3 +3X4 +3X5 +4X6 = 1  
3X1 -2X2 -1X3 +4X4 +3X5 -3X6 = 2  
4X1 -2X2 +1X3 -1X4 -1X5 -1X6 = 2  
2X1 +3X2 +3X3 +1X4 +2X5 -3X6 = 3   
  
Вводим в ограничения [искусственные переменные](http://www.tisbi.ru/resources/lib/linprog/theory3.htm#isk) :   
  
-3X1 +0X2 +2X3 +3X4 +3X5 +4X6 +X7 = 1  
3X1 -2X2 -X3 +4X4 +3X5 -3X6 +X8 = 2  
4X1 -2X2 +X3 -X4 -X5 -X6 +X9 = 2  
2X1 +3X2 +3X3 +X4 +2X5 -3X6 +X10 = 3   
  
Вводим в целевую функцию дополнительные слагаемые:  
  
F(X) = 4X1 +4X2 +2X3 +4X4 +3X5 +X6 - MX7 - MX8 - MX9 - MX10 => max   
  
Т.о. мы получили [новую ЗЛП](http://www.tisbi.ru/resources/lib/linprog/theory3.htm#newz).   
  
Сформируем [начальное базисное решение](http://www.tisbi.ru/resources/lib/linprog/theory3.htm#nbaz) новой М-задачи :   
  
X'=( 0, 0, 0, 0, 0, 0, 1, 2, 2, 3 );   
  
После решения М-задачи симплекс методом мы получили следующее оптимальное решение : X"=( 1.2, 0.77, 0.00, 0.54, 0.00, 0.75, 0.00, 0.00, 0.00, 0.00 ); В этом решении все искусственные переменные равны 0, следовательно мы нашли оптимальное решение и для исходной ЗЛП : X = ( 1.2, 0.77, 0.00, 0.54, 0.00, 0.75 ).

1.6. Элементы теории двойственности

Каждой задаче ЛП соответствует некоторая другая задача, которую называют двойственной (сопряженной) к исходной.

Предположим, что исходная задача представлена в стандартной форме:



 (1.8)

Задачей ЛП, *двойственной* к задаче (8), называется задача, представленная в виде:



 (1.9)

Обратно, если рассматривается задача (1.9), то задача (1.8) называется двойственной к (1.9).

Отметим основные соотношения между двойственными задачами:

* количество переменных в двойственной задаче равно количеству основных ограничений неравенств;
* коэффициенты целевой функции одной из задач являются свободными членами в ограничениях двойственной задачи ;
* если первая задача (1.8) является задачей на максимум целевой функции, то задача (1.9) - задачей на минимум целевой функции;
* матрицы коэффициентов перед переменными в левых частях ограничений-неравенств обеих двойственных задач являются транспонированными друг к другу;
* знаки в основных ограничениях-неравенствах меняются на противоположные.

*Пример 1.7.* Записать задачу, двойственную к задаче:





*Решение.* Прежде всего, приведем исходную задачу к стандартному виду (1.8). Для этого умножим обе части второго неравенства на (-1). Получим новую систему ограничений:



Данная система содержит два основных ограничения-неравенства. Поставим им в соответствие переменные y1 и y2. Запишем целевую функцию двойственной задачи; используя в качестве ее коэффициентов свободные члены 8 и -6:



Матрица коэффициентов перед переменными в левых частях основных ограничений имеет вид:



Транспонируя матрицу А, запишем ограничения-неравенства двойственной задачи:

 (1.11)

Свободные члены ограничений-неравенств соответствуют коэффициентам целевой функции L.

Пара двойственных задач связана определенными соотношениями, составляющими предмет теории двойственности.

# Основная теорема двойственности

Если одна из двойственных задач имеет оптимальное решение, то вторая задача также имеет оптимальное решение, причем экстремальные значения целевых функций на этих решениях совпадают:

.

Если же одна из задач неразрешима из-за неограниченности целевой функции ( или ), то допустимое множество решений второй задачи пусто.

Из двух задач (1.8) и (1.9) первая решается симплекс-методом обычно легче.

Если в (1.8) все , то, приводя исходную задачу к основной форме, получим следующую систему ограничений:



Тем самым, сразу становится известен первый опорный план:



и, соответственно, первая симплекс-таблица:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Базис* | *x1* | *…* | *xn* | *xn+1* | *…* | *xn+m* | *bi* |
| *xn+1* | *a11* | *…* | *a1n* | 1 | *…* | *0* | *b1* |
| *…* | *…* | *…* | *…* | *…* | *…* | *…* | *…* |
| *xn+m* | *am1* | *…* | *amn* | *0* | *…* | 1 | *bm* |
|  | *-c1* | *…* | *-cn* | *0* | *…* | *0* |

Предположим, что данная задача имеет единственное решение. Можно показать, что последняя симплекс-таблица, по которой был рассчитан оптимальный план задачи , определяет также и оптимальный план двойственной задачи. Его компоненты  находятся в правом нижнем углу этой симплекс-таблицы в столбцах, соответствующих добавленным компонентам :

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| *Базис* | *x1* | *…* | *xn* | *xn+1* | *…* | *xn+m* | *bi* |
| *…* | *…* | *…* | *…* | *…* | *…* | *…* | *…* |
|  | *…* | *…* | *..* |  | *…* |  |

# Вторая теорема двойственности

Для того, чтобы два допустимых решения  и  были оптимальными решениями пары двойственных задач, необходимо и достаточно, чтобы они удовлетворяли системе уравнений:





Третья теорема двойственности

Значения компонент в оптимальном решении двойственной задачи представляет собой коэффициенты чувствительности к изменениям значений свободных членов в исходной задаче.

Иначе говоря, пусть наряду с задачей ЛП (1.8) рассматривается также задача вида:



 (1.12)

Пусть также  и  - оптимальные значения целевых функций соответственно задач (1.8) и (1.12),  - оптимальный план задачи (1.9). Тогда значения  и  связаны соотношением:

 (1.13)

Методы решения целочисленных ЗЛП.

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 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 задача является задачей линейного программирования.

Методы решения задач целочисленного программирования можно классифицировать как методы отсечений (1) и комбинаторные методы (2).

Исходной задачей для методов отсечений, используемых при решении линейных целочисленных задач, является задача с ослабленными ограничениями, которая возникает в результате исключения требования целочисленности переменных. По мере введения специальных дополнительных ограничений, учитывающих требования целочисленности, многогранник допустимых решений ослабленной задачи постепенно деформируется до тех пор, пока координаты допустимого решения не станут целочисленными. Название «методы отсечений» связано с тем обстоятельством, что вводимые дополнительные ограничения отсекают (исключают) некоторые области многогранника допустимых решений, в которых отсутствуют точки с целочисленными координатами,

В основе комбинаторных методов лежит идея перебора всех допустимых целочисленных решений, разумеется, на первый план здесь выдвигается проблема разработки тестовых процедур, позволяющих непосредственно рассматривать лишь относительно небольшую часть указанных решений, а остальные допустимые решения учитывать некоторым косвенным образом. Наиболее известным комбинаторным методом является метод ветвей и границ, который также опирается на процедуру решения задач с ослабленными ограничениями. При таком подходе из рассматриваемой задачи получаются две подзадачи путем специального «разбиения» пространства допустимых решений и отбрасывания областей, не содержащих допустимых целочисленных решений.

В случае, когда целочисленные переменные являются булевыми, применяются комбинированные методы. Булевы свойства переменных существенно упрощают поиск решения.

Алгоритм метода отсечений для решения полностью целочисленной задачи.

Необходимым условием применения данного алгоритма является целочисленность всех коэффициентов и правых частей ограничений исходной задачи. Любое ограничение с рациональными коэффициентами легко приводится к требуемому виду путем умножения ограничения на наименьший общий знаменатель входящих в него коэффициентов.

Алгоритм состоит в следующем. На первом шаге решается задача с ослабленными ограничениями, не содержащая условий целочисленности переменных. Если полученное оптимальное решение оказывается целочисленным, то оно является также решением исходной задачи. В противном случае следует ввести дополнительные ограничения, порождающие (вместе с некоторыми ограничениями) новую задачу линейного программирования, решение которой оказывается целочисленным и совпадает с оптимальным решением исходной целочисленной задачи. Пусть последняя симплекс-таблица задачи с ослабленными ограничениями имеет следующий вид:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Базисные переменные | x1 | … | xi | … | xm | w1 | … | wj | … | wn | Решение |
| Z | 0 | … | 0 | … | 0 | C1 | … | Cj | … | Cn | b0 |
| x1 | 1 | … | 0 | … | 0 | a11 | … | aj1 | … | an1 | b1 |
| … | … | … | … | … | … | … | … | … | … | … | … |
| xi | 0 | … | 1 | … | 0 | a1i | … | aji | … | ani | bi |
| ... | … | … | … | … | … | … | … | … | … | … | … |
| xm | 0 | … | 0 | … | 1 | a1m | … | ajm | … | anm | bm |

Рассмотрим i-ую строку, которой соответствует нецелое значение базисной переменной xi, и выразим xi через небазисные переменные:

http://www.bestreferat.ru/images/paper/55/07/9040755.png, bI – нецелое.

Каждую строку симплекс-таблицы, порождающую аналогичное равенство будем называть производящей строкой. Так как коэффициенты целевой функции можно считать целыми числами, переменная Z также должна быть целочисленной, и верхняя строка таблицы также может быть выбрана в качестве производящей. Пусть

bI=[bI]+fi, aji=[aji]+fij, 0<fi<1, 0£fij<1.

В качестве дополнительного ограничения вводим такое

http://www.bestreferat.ru/images/paper/56/07/9040756.png,

где Si – неотрицательная дополнительная переменная, которая по определению должна принимать целые значения. Такое ограничение равенство определяет отсечение Гомори для полностью целочисленной задачи. Добавив построенное ограничение в симплекс-таблицу, получим недопустимое, но оптимальное решение. В такой ситуации следует использовать двойственный симплекс-метод для получения допустимого и оптимального решения.

Метод ветвей и границ.

На первом шаге также решается задача с ослабленными ограничениями, не содержащая условий целочисленности переменных. Но в отличие от методов отсечений этот метод может применяться как для полностью целочисленной задачи, так и для частично целочисленной. Если полученное оптимальное решение оказывается целочисленным, то оно является также решением исходной задачи. Идея метода заключается в следующем. Пусть xr целочисленная переменная, значение которой xr в оптимальном решении ослабленной задачи является дробным. Интервал

[xr]< xr<[xr]+1

не содержит допустимых целочисленных компонент решения. Поэтому допустимое целое значение xr должно удовлетворять одному из неравенств [xr]³ xr или xr³ [xr]+1. Введение этих условий в задачу с ослабленными ограничениями порождает две не связанные между собой задачи. В таком случае говорят, что исходная задача разветвляется на две подзадачи. Осуществляемый в процессе ветвления учет необходимых условий целочисленности позволяет исключить части многогранника допустимых решений, не содержащие точек с целыми координатами.

Затем каждая из подзадач решается как задача линейного программирования двойственным симплекс-методом. Если полученный оптимум оказывается допустимым для целочисленной задачи, то это решение следует зафиксировать как наилучшее. В противном случае подзадача в свою очередь должна быть разбита на две подзадачи по другой переменной и т.д.

Задача линейного программирования транспортного типа.

Постановка задачи. Пусть в m пунктах производят некоторый однородный продукт, причем объем производства в пункте i составляет Ai единиц. Допустим, что данный продукт потребляется в n пунктах потребления, а объем потребления в пункте j составляет единиц Bj. Предположим, что из каждого пункта производства i возможна транспортировка продукта в любой пункт потребления j с затратами cij. Задача состоит в определении такого плана перевозок, при котором запросы всех потребителей полностью удовлетворены, весь продукт вывезен из пунктов производства и суммарные транспортные издержки минимальны.

## Математическая модель. Пусть xij – количество продукта, перевозимого из пункта i в пункт j. Найти множество переменных удовлетворяющих условиям

## http://www.bestreferat.ru/images/paper/57/07/9040757.png,

http://www.bestreferat.ru/images/paper/58/07/9040758.png.

## и таких, что целевая функция

## http://www.bestreferat.ru/images/paper/59/07/9040759.png

## достигает минимума.

Метод решения транспортной задачи [6, 7, 10].

##### Сети

##### Многие задачи линейного программирования можно сформулировать и решить с помощью сетевых моделей. Специальная структура этих задач позволяет разработать эффективные алгоритмы, которые в большинстве случаев основываются на теории линейного программирования.

##### Задача минимизации сети.

##### Задача минимизации сети состоит в нахождении ребер, соединяющих все узлы сети и имеющих минимальную суммарную длину. Для решения задачи необходимо построить минимальное дерево-остов, применяя следующий итеративный процесс. Начать с любого узла и соединить его с ближайшим узлом сети. Соединенные узлы образуют теперь связное множество, а остальные узлы – несвязное. Далее в несвязном множестве выбрать узел, расположенный ближе всего к одному из узлов связного множества. Скорректировать соответствующим образом связное и несвязное множества, а дугу, по которой произошло присоединение запомнить. Процесс повторять до тех пор, пока все узлы не окажутся в связном множестве. Выбранные дуги образуют минимальное дерево-остов. Его длина равна сумме длин этих дуг.

### Задача о кратчайшем пути

### Задача о кратчайшем пути состоит в нахождении связанных между собой дорог на транспортной сети, которые имеют минимальную длину от исходного пункта до пункта назначения. Для решения этой задачи можно применить следующий алгоритм. Каждому узлу сети будем приписывать временные пометки равные расстоянию от начального узла до данного узла. Если оказывается, что узел принадлежит кратчайшему маршруту, то временную пометку объявляем постоянной. На первой итерации начальному узлу приписывается постоянная пометка равная нулю, а остальным узлам – временные пометки, равные длине дуги из начального узла в рассматриваемый узел, если такая дуга существует и «¥», если нет такой дуги. Затем, до тех пор пока конечный узел не получит постоянную пометку выполняются следующие две процедуры: 1) среди временных пометок выбирается минимальная и объявляется постоянной; 2) для всех временно помеченных узлов вычисляются новые временные пометки, меньшей из двух величин – старой временной пометки рассматриваемого узла и суммы постоянной пометки последнего постоянно помеченного узла и длины дуги, соединяющей последний постоянно помеченный узел с рассматриваемым узлом. Если при этом постоянную пометку получает конечный узел, то кратчайший маршрут найден. Дуги входящие в этот маршрут определяются следующим образом: если разность между постоянными пометками начального и конечного узлов данной дуги равна длине дуги, то эта дуга принадлежит кратчайшему маршрут.

### Задача о максимальном потоке

### Рассмотрим задачу о максимальном потоке между двумя выделенными узлами связной сети. Каждая дуга сети обладает пропускными способностями в обоих направлениях, которые определяют максимальное количество потока, проходящего поданной дуге. Ориентированная (односторонняя) дуга соответствует нулевой пропускной способности в запрещенном направлении.

### Пропускные способности cij сети можно представить в матричной форме. Для определения максимального потока из источника s в сток t используется следующий алгоритм.

### Шаг 1. Найти цепь, соединяющую s с t, по которой поток принимает положительное значение в направлении s®t. Если такой цепи не существует, перейти к шагу 3. В противном случае перейти к шагу 2.

### Шаг 2. Пусть cij- (cij+) – пропускные способности дуг цепи (s, t) в направлении s®t (t®s) и q = min{cij-}>0. Матрицу пропускных способностей (cij) изменить следующим образом:

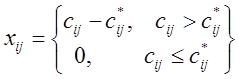
### (а) вычесть q из всех cij- ;

### (б) прибавить q ко всем cij+ .

### Заменить текущую cij-матрицу на вновь полученную и перейти к шагу 1.

### Операция (а) дает возможность использовать остатки пропускных способностей дуг выбранной цепи в направлении s®t. Операция (б) восстанавливает исходные пропускные способности сети, поскольку уменьшение пропускной способности дуги в одном направлении можно рассматривать как увеличение ее пропускной способности в противоположном направлении.

### Шаг 3. Найти максимальный поток в сети. Пусть C = ççcijçç - исходная матрица пропускных способностей, и пусть C\* = ççcijçç - последняя матрица, получившаяся в результате модификации исходной матрицы (шаги 1 и 2). Оптимальный поток X = ççxijçç в дугах задается как



### Максимальный поток из s в t равен

### http://www.bestreferat.ru/images/paper/61/07/9040761.png

### При этом z есть сумма всех положительных q, определенных на шаге 2. Таким образом, можно объяснить, почему используются положительные элементы матрицы C – C\* для определения результирующего потока в направлении s® t.