**ФЕДЕРАЛЬНОЕ АГЕНТСТВО ВОЗДУШНОГО ТРАНСПОРТА**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ГРАЖДАНСКОЙ АВИАЦИИ (МГТУ ГА)»**

**Кафедра вычислительных машин, комплексов, систем и сетей**

Е.В. Юркевич

# МЕТОДЫ ОПТИМИЗАЦИИ

## Ч. II ОПТИМАЛЬНЫЕ ПРОЦЕССЫ

**Учебно-методическое пособие**
по выполнению практических работ

*для студентов*
*направления 09.03.01*
*очной формы обучения*

Москва
2019

ББК 6Ф7.3
Ю74

Рецензент:
*Романчева Н.И.* – канд. техн. наук, доц. каф. ВМКСС

**Юркевич Е.В.**
Ю74 Методы оптимизации Ч.II Оптимальные процессы: учебно-методическое пособие по выполнению практических работ./ Е.В. Юркевич. – Воронеж: ООО «МИР», 2019. – 32 с.

Данное учебно-методическое пособие издается в соответствии с рабочей программой учебной дисциплины «Методы оптимизации» по учебному плану для студентов направления 09.03.01 очной формы обучения.

Рассмотрено и одобрено на заседании кафедры 26.02.2019 г. и методического совета 26.02.2019 г.

*В авторской редакции*

Подписано в печать 25.03.2019 г.
Формат 60x84/16 Печ.л. 2 Усл. печ. л. 1,86
Заказ 444/090451 Тираж 30 экз.

Московский государственный технический университет ГА
*125993 Москва, Кронштадтский бульвар, д.20*
Отпечатано ООО «МИР»
*394033, г. Воронеж, Ленинский пр-т 119А, лит. Я, оф.215*

**С О Д Е Р Ж А Н И Е**

Основные требования и порядок выполнения практических работ 3

1 ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 6.**Исследование операций Математическое программирование. Линейное программирование** 4

1.1 Цель занятия 4

1.2 Задачи по теме занятия 4

1.3 Основные теоретические сведения 4

1.4 Контрольные вопросы к защите занятия 12

1.5 Список рекомендуемой литературы 12

2 ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № **7. Целочисленное программирование** 13

2.1 Цель занятия 13

2.2 Задачи по теме занятия 13

2.3 Основные теоретические сведения 13

2.4 Контрольные вопросы к защите занятия 18

2.5 Список рекомендуемой литературы 18

3 ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 8.**Нелинейное программирование** 18

3.1 Цель практической работы 18

3.2 Задачи по теме занятия 18

3.3 Основные теоретические сведения 18

3.4 Контрольные вопросы к защите занятия 22

3.5 Список рекомендуемой литературы 22

4 ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 9. **Стохастическое программирование** 22

4.1 Цель занятия 22

4.2 Задачи по теме занятия 22

4.3 Основные теоретические сведения 22

4.4 Контрольные вопросы к защите занятия 25

4.5 Список рекомендуемой литературы 25

5 ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 10. **Динамическое программирование** 25

5.1 Цель занятия 25

5.2 Задачи по теме занятия 25

5.3 Основные теоретические сведения 26

5.4 Контрольные вопросы к защите занятия 30

5.5 Список рекомендуемой литературы 30

# ОСНОВНЫЕ ТРЕБОВАНИЯ И ПОРЯДОК ВЫПОЛНЕНИЯ ПРАКТИЧЕСКИХ РАБОТ

Настоящее учебно-методическое пособие предназначено для студентов направления подготовки «Информатика и вычислительная техника» (бакалавриат), выполняющих практические задания по дисциплине «Методы оптимизации». В пособие включены материалы по практическим занятиям № 6, 7, 8, 9, 10.

Целью проведения практических занятий является закрепление основных теоретических положений, изложенных в лекциях.

Практическая работа состоит из следующих этапов:

1) домашняя подготовка;
2) выполнение работы в соответствии с заданием;
3) сдача выполненной работы преподавателю.
4) распечатка результатов работы на принтере;
5) защита практической работы.

В процессе домашней подготовки студент:

- изучает лекционный материал, материалы по темам данного пособия и дополнительной литературы,
- знакомится с заданием на выполнение практической работы;
- готовит отчет по выполнению практической работы (пункты со знаком *).

ВЫПОЛНЕНИЕ практической работы производится во время занятий в присутствии преподавателя. В процессе выполнения практической работы студент последовательно выполняет задание. По завершению работы – демонстрирует преподавателю результаты.

СДАЧА РАБОТЫ преподавателю заключается в демонстрации выполненной работы и выполнении непосредственно при преподавателе индивидуального задания.

ОТЧЕТ по каждой практической работе должен содержать:

- название работы*;
- цель практической работы*;
- задание на выполнение практической работы*;
- краткие комментарии по выполнению практической работы*;
- распечатки файлов результатов, подписанные преподавателем.

ЗАЩИТА практической работы преподавателю проводится по контрольным вопросам и при наличии оформленного отчета (распечатки должны быть приклеены). После защиты практической работы делается соответствующая запись на отчете студента.

# 1. ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 6. ИССЛЕДОВАНИЕ ОПЕРАЦИЙ. МАТЕМАТИЧЕСКОЕ ПРОГРАММИРОВАНИЕ. ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

## 1.1 Цель практического занятия:

знакомство с задачами оптимизации технологических процессов.

## 1.2 Задачи по теме занятия:

закрепить знания по разделам:

- исследование операций и системный анализ сложных процессов;
- математическое программирование технологического процесса;
- понятие об оптимальном процессе;
- линейное программирование;
- стандартная и каноническая формы задачи линейного программирования;
- симплекс метод;
- двойственная задача.

## 1.3 Основные теоретические сведения

**Определение.** *Исследование операций - область прикладной математики, в которой используются количественные методы для оценки эффективности организационной и управленческой деятельности с целью выбора оптимальных или близких к оптимальным решений для достижения поставленной цели.*

Использование исследования операций в проектировании средств и систем начинается с построения математической модели разработки проекта (исследуемой операции). Для этого требуется точно описать цель (цели) операции, указать возможные (допустимые) действия или решения и количественно оценить их результаты. Обычно такое описание представляется в виде

$$E=f(x,y), x \in X \in R_n, y \in Y \in R_m, E=f(x,y), x \in X \in Rn, y \in Y \in Rm,$$

$$E=f(x,y), x \in X \in R_n, y \in Y \in R_m,$$

где элементами $\{x\}$ множества $X$ являются допустимые действия, т. е. управляемые переменные, элементы $\{y\}$ множества $Y$ являются параметры модели, т. е. неуправляемые переменные, $E$ – результат принятия решения ($x \in X$) при фиксированном значении параметра ($y \in Y$). Функция $f$, определяющая численное значение результата решения, называется целевой функцией (показателем эффективности). На этом этапе одну из основных задач выбора модели системообразующего параметра (параметров) который обеспечивает получение оптимального значения показателя эффективности $E$, который должен быть представлен руководителю операции для принятия окончательного решения.

Поскольку, как правило, модель операции содержит неизвестные параметры, решение принимается в условиях неопределённости. Часто параметры модели представляют собой случайные величины, поэтому теория вероятностей и математическая статистика играют в важную роль как при построении модели, так и при проверке её адекватности.

В задачах исследования операций используются математические методы, общие для прикладных задач: математический анализ, линейная алгебра, дифференциальные уравнения, теория вероятностей. Специфическими являются методы оптимизации, в частности линейное программирование, динамическое программирование, а также такие разделы теории вероятностей, как теория массового обслуживания и статистическое моделирование. Если в задаче имеется не одна, а несколько целевых функций, то для её решения привлекается теория многокритериальной оптимизации.

В организации разработок технических средств используются принципы системного анализа, определяющие постановку и решения задач по улучшению функционирования

реальных производств, обслуживающих и управляющих систем, а также способы реализации выработанных рекомендаций с учётом наиболее существенных для этих систем последствий. Исследование операций и, в частности, математическое программирование являются исходным пунктом развития ряда научных направлений, ориентированных на разработку методологии прикладных исследований, прежде всего *системного анализа как математического выражения здравого смысла.*

**Определение.** *Математическое программирование – один из разделов прикладной математики, использующий методы системного анализа и являющийся инструментом для поиска оптимальных решений, т. е. выбора оптимальной программы ведения технологического процесса.* Под принятием решения понимается технология, в которой выделяют этапы:

1-й этап. Постановка проблемы и формулировка цели её решения.

2-й этап. Построение математической модели объекта (процесса), в виде системы с качественным представлением факторов, определяющих значения системообразующих параметров, влияющих на достижение поставленной цели. Обычно модель строится с использованием статистических методов.

3-й этап. Нахождение оптимального решения, т. е. расчет *max (min)* целевой функции в соответствии с закономерностями, определяющими существо поставленной проблемы. Такие закономерности рассматриваются как ограничения в нахождении оптимального решения. Оптимизация может быть условной и безусловной. Условная оптимизация рассматривается в пределах заданных ограничений, безусловная предполагает нахождение экстремальных значений системообразующих параметров на всем физически возможном диапазоне параметров

Практически управление предполагает решение экстремальных задач, где ограничения записываются в виде равенств и неравенств в значениях параметров, входящих в модель. Теория и методы решения этих задач составляют содержание математического программирования.

4-й этап. В соответствии с технологией моделирования на этом этапе устанавливается адекватность модели и моделируемого объекта в пределах точности исходной информации, определенной в техническом задании на разработку данного объекта (процесса). Сопоставление результатов вычислений, полученных на 3-м этапе, с характеристиками моделируемого объекта является результатом валидации сформированной модели.

Методы оптимизации разделяются на две группы: безградиентные, использующие только значения целевой функции, и градиентные, базирующиеся на использовании производных минимизируемой функции. Безградиентные методы (например, метод сеток) более пригодны для задач, где минимизируемая функция существенно нелинейна или имеет разрывы. Градиентные методы первого порядка, с использованием первой производной (например, метод Зейделя) обычно эффективны в случаях целевых функций, непрерывных вместе с первыми производными. Методы второго порядка, с использованием второй производной (например, метод Ньютона), применяются реже, поскольку требуют больших вычислительных затрат для расчета матриц вторых производных. Такие методы используют информацию о наклоне функции для выбора направления поиска экстремума.

В целом, в математическом программировании выделяют направления:

- решение детерминированных задач, предполагающее, что исходная информация является полностью определенной. К этому направлению относятся линейное программирование, нелинейное программирование, особым направлением является целочисленное программирование;

- стохастическое программирование – решение задач, в которых исходная информация содержит элементы с вероятностными характеристиками, либо некоторые параметры задачи носят случайный характер. Например, задача оптимизации надежности работы автоматического устройства при воздействии случайных помех. Одна из главных

трудностей стохастического программирования состоит в самой постановке задач, главным образом из-за сложности анализа исходной информации;

- решение задач, в которых исходная информация характеризуется нечисловыми (качественными) оценками, действия с которыми не подчиняются законам классической (цифровой) математики. Например, требуется оптимизировать технологическое оснащение производства, использующего нескольких агрегатов. Известны качественные оценки удобства работы с каждым из агрегатов в отдельности. Требуется максимизировать удобство работы оператора, управляющего всеми агрегатами в целом.

**Определение.** *Оптимальным называется процесс, при котором целевая функция достигает экстремального значения (максимума или минимума). Формирование такой целевой функции в качестве модели процесса определяется существом задачи.*

**Определение.** *Линейное программирование (ЛП) – раздел математического программирования, где целевая функция линейна, и множество, на котором ищется экстремум целевой функции, задается системой линейных равенств и неравенств.*

В ЛП существуют классы задач, структура которых позволяет создать специальные методы их решения, выгодно отличающиеся от методов решения задач общего характера (транспортные задачи, задачи о рюкзаке и т.д.) Важным разделом является целочисленное программирование, когда на переменные накладываются условия целочисленности (оптимизация выпуска изделий или их транспортировки).

Математическая модель задач ЛП включает в себя:

- представление критерия оптимальности (*max* или *min* целевой функции);
- систему ограничений в форме линейных уравнений и неравенств;
- требование неотрицательности переменных.

Линейная форма $<c,x> = c_1x_1 + c_2x_2 + \ldots + c_nx_n$, подлежащая максимизации (или минимизации), рассматривается как целевая функция. Вектор $X=(x_1,x_2,\ldots,x_n)$, удовлетворяющий всем ограничениям задачи ЛП, называется допустимым вектором, или допустимым планом. Область, определенная системой ограничений, является областью допустимых векторов (допустимых планов). Допустимый вектор $X^*$, доставляющий наибольшее значение целевой функции по сравнению с любым другим допустимым вектором $X$, т.е. $< c, x^* > \geq < c, x >$, называется решением задачи, или оптимальным планом. Максимальное значение $d = < c, x^* >$ целевой функции называется значением задачи.

В зависимости от типа ограничений различают три формы задач ЛП

Стандартная задача ЛП.

$$\begin{aligned} &\max (c_1x_1 + c_2x_2 + \ldots + c_nx_n) \\ &a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1, \\ &a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2, \\ &\ldots \\ &a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leq b_m, \\ &x_1 \geq 0, x_2 \geq 0, \ldots x_n \geq 0, \end{aligned}$$

или, в матричной записи,

$$\begin{aligned} &\max < c, x > \\ &Ax \leq b, x \geq 0, \end{aligned}$$

где $x = (x_1, x_2, \ldots, x_n), c = (c_1, c_2, \ldots, c_n), b = (b_1, b_2, \ldots, b_m), A = (a_{ij})$ – матрица коэффициентов. Вектор $c$ называется вектором коэффициентов линейной формы, $b$ – вектором ограничений.

Стандартная задача важна ввиду наличия большого числа прикладных моделей, сводящихся к этому классу задач ЛП.

Каноническая задача ЛП.

$$\max(c_1x_1 + c_2x_2 + \ldots + c_nx_n)$$
$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1,$$
$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2,$$
$$\ldots$$
$$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m,$$
$$x_1 \geq 0, x_2 \geq 0, \ldots x_n \geq 0,$$

или, в матричной записи,

$$\max < c, x >$$
$$Ax = b, x \geq 0,$$

Общая задача ЛП. В этой интерпретации задачи часть ограничений носит характер неравенств, а часть является уравнениями. При этом, не на все переменные наложено условие неотрицательности:

$$\max(c_1x_1 + c_2x_2 + \ldots + c_nx_n)$$
$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1,$$
$$a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2,$$
$$\ldots$$
$$a_{k1}x_1 + a_{k2}x_2 + \ldots + a_{kn}x_n \leq b_k,$$
$$a_{k+1,1}x_1 + a_{k+1,2}x_2 + \ldots + a_{k+1,n}x_n = b_{k+1},$$
$$a_{k+2,1}x_1 + a_{k+2,2}x_2 + \ldots + a_{k+2,n}x_n = b_{k+2},$$
$$\ldots$$
$$a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m,$$
$$x_1 \geq 0, x_2 \geq 0, \ldots x_r \geq 0,$$

Здесь $k \leq m$, $r \leq n$ стандартная задача образуется при $k = m$. $r = n$;
каноническая – при $k = 0, r = n$.

Правило приведения задачи ЛП к каноническому виду:

1. Если в исходной задаче некоторое ограничение (например, первое) было неравенством, то оно преобразуется в равенство, введением в левую часть некоторой неотрицательной переменной, при чем в неравенства «≤» вводится дополнительная неотрицательная переменная со знаком «+»; в случаи неравенства «≥» – со знаком «-». Например:

$$a_{11}x_1 + a_{12}x_1 + \ldots + a_nx_n \leq b_1 \quad (1)$$

Вводим переменную $x_{n+1} = b_1 - a_{11}x_1 - a_{11}x_1 - \ldots - a_{1n}x_n$,
тогда неравенство (1) запишется в виде:

$$a_{11}x_1 + a_{12}x_1 + \ldots + a_nx_n + x_{n+1} = b_1.$$

В каждое из неравенств вводится своя «уравнивающая» переменная, после чего система ограничений становится системой уравнений.

2. Если в исходной задаче некоторая переменная не подчинена условию неотрицательности, то ее заменяют (в целевой функции и во всех ограничениях) разностью неотрицательных переменных $x_k = x_k - x_l$, $x_k \geq 0$, $x_l \geq 0$, $l$ - свободный индекс.

3. Если в ограничениях правая часть отрицательна, то следует умножить это ограничение на (-1)

4. Если исходная задача была задачей на *min*, то введением новой целевой функции $F_1 = -F$ задачу на *min* функции $F$ преобразуют в задачу на *max* функции $F_1$.

Таким образом, всякую задачу ЛП можно сформулировать в канонической форме.

**Двойственная задача линейного программирования**

Двойственную задачу именуют «Задачей разборчивой невесты», желающей получить «самый лучший подарок», но, когда у неё будет «самое лучшее настроение». В

математической постановке, такая задача интерпретируется как максимизация характеристик процесса по двум критериям. Сначала формулируется «прямая» задача (2),

$$\begin{aligned} &\max(c_1x_1 + c_2x_2 + \ldots + c_nx_n) \\ &a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \le b_1, \\ &a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \le b_2, \\ &\ldots \\ &a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \le b_m, \end{aligned} \qquad (2)$$

или, в матричной записи,

$$\begin{aligned} &\max < c, x > \\ &Ax \le b. \end{aligned} \qquad (3)$$

Задачей, двойственной к (2) (двойственной задачей), называется задача ЛП от $m$ переменных $p_1, p_2, \ldots, p_m$ вида

$$\begin{aligned} &min\ (b_1p_1 + b_2p_2 + \ldots + b_mp_m) \\ &a_{11}p_1 + a_{12}p_2 + \ldots + a_{1m}p_m = c_1 \\ &a_{21}p_1 + a_{22}p_2 + \ldots + a_{2m}p_m = c_2 \\ &\quad\ldots\ldots \\ &a_{n1}p_1 + a_{n2}p_2 + \ldots + a_{nm}p_m = c_n \end{aligned} \qquad (4)$$

или, в матричной записи,

$$\begin{aligned} &\min\ < b, p > \\ &A'p = c, p \ge 0, \end{aligned} \qquad (5)$$

где $p = (p_1, p_2, \ldots, p_m)$.

Правила построения задачи (4) по форме записи задачи (2) таковы: в задаче (4) переменных $p_i$ столько же, сколько строк в матрице $A$ задачи (2). Матрица ограничений в (4) - транспортированная матрица $A$. Вектор правой части ограничений в (4) служит вектором коэффициентов максимизируемой линейной формы в (2), при этом знаки неравенств меняются на равенство. Наоборот, в качестве целевой функции в (4) выступает линейная форма, коэффициентами которой задаются вектором правой части ограничений задачи (2), при этом максимизация меняется на минимизацию. На двойственные переменные $p_i$ накладывается условие неотрицательности.

**Теорема двойственности**. *Если взаимодвойственные задачи (3), (5) допустимы, то они обе имеют решение и одинаковое значение*.

**Теорема равновесия**. *Пусть* $x^* = (x_1^*, x_2^*, \ldots, x_n^*)$, $p^* = (p_1^*, p_2^*, \ldots, p_m^*)$ *– оптимальные планы прямой (2) и двойственной (4) задач соответственно. Тогда, если* $p_i^* > 0$, *то* $a_{i1}x_1^* + a_{i2}x_2^* + \ldots + a_{in}x_n^* = b_i$

**Симплекс метод –** *универсальный метод решения задач ЛП. Симплексом называется многогранник, образованный системой ограничений, определяющих область допустимых планов согласно постановке задачи ЛП.*

*Идея метода.* Исходя из практических особенностей решаемой задачи, выбрать начальную вершину симплекса, т.е. вершину многогранника области допустимых планов. Она будет рассматриваться как начальное допустимое решение. Далее требуется проверить это решение на оптимальность. Если оно оптимально, то решение найдено. Если – нет, то требуется перейти к другой вершине многогранника и проверить на оптимальность её. Ввиду конечности вершин многогранника (конечности ограничений в задаче ЛП) за конечное число «шагов» определяется искомый план (максимальный или минимальный). В практических расчетах следует учитывать, что при переходе от вершины к вершине

желательно, чтобы значение целевой функции убывало (при задаче на минимум) и возрастало (при задаче на максимум).

**Пример.** Для производства двух видов изделий А и В используют три типа технологического оборудования. Для производства единицы изделия А оборудование первого типа используется в течении 1 часа, оборудование второго типа – 3 часа, оборудование третьего типа – 3 часа.

Для производства единицы изделия В оборудование первого типа используется в течении 2 часа, оборудование второго типа – 3 часа, оборудование третьего типа – 1 час. На изготовление всех изделий предприятие может использовать оборудование первого типа не более чем 32 часа, оборудование второго типа – 60 часов, оборудование третьего типа – 50 часов.

Прибыль от реализации единицы готового изделия А составляет 4 денежные единицы, а изделия В – 2 денежные единицы.

*Задание.* Составить план производства изделий А и В, обеспечивающий максимальную прибыль от их реализации. Для этого составим математическую модель задачи и решим её графическим методом и симплекс-методом путем преобразования симплекс-таблиц.

*Решение.* Под планом производства понимается ответ на простой вопрос: сколько изделий А и сколько изделий В надо выпустить, чтобы прибыль была максимальна.

Прибыль рассчитывается по формуле: $F = 4x_1 + 2x_2$.

Математическая модель задачи:

$$\begin{cases} x_1 + 2x_2 \le 32 \\ 3x_1 + 3x_2 \le 60 \\ 3x_1 + x_2 \le 50 \end{cases}$$

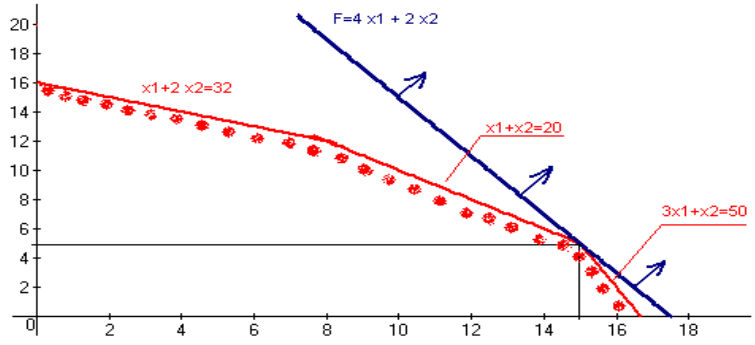
$$x_1 \ge 0 \quad x_2 \ge 0$$

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

Чтобы проиллюстрировать применение симплекс-метода решения этой задачи, решим ее графически. Для этого построим на плоскости ($x_1$, $x_2$) области, описываемые ограничениями-неравенствами, и прямую $F=4x_1+2x_2$, которая является целевой функцией.

Три записанных выше неравенства ограничивают на плоскости многоугольник (линия с точками), ограниченный слева и снизу координатными осями (т.к. искомое количество изделий положительно). График целевой функции (линия со стрелками) передвигается в направлении, обозначенном стрелкой (в направлении своего градиента), до тех пор, пока не достигнет граничной точки многоугольника – в нашем случае это точка – (15; 5). В этой точке целевая функция будет достигать максимума.

$$F(15;5) = 60 + 10 = 70$$




Решим эту задачу *симплекс-методом*. Для этого перейдем от ограничений-неравенств к ограничениям-равенствам, введя дополнительные переменные $x_3, x_4, x_5 \geq 0$.

$$\begin{cases} x_1 + 2x_2 + x_3 = 32 \\ x_1 + x_2 + x_4 = 20 \\ 3x_1 + x_2 + x_5 = 50 \end{cases}$$

$x_i \geq 0 \quad i = 1..5$

$F(x_1, x_2) = 4x_1 + 2x_2 + 0x_3 + 0x_4 + 0x_5 \rightarrow \max$

| $i$ | Базис | $C_б$ | $B$ | 4 | 2 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | $A_1$ | $A_2$ | $A_3$ | $A_4$ | $A_5$ |
| 1 | $A_3$ | 0 | 32 | 1 | 2 | 1 | 0 | 0 |
| 2 | $A_4$ | 0 | 20 | 1 | 1 | 0 | 1 | 0 |
| 3 | $A_5$ | 0 | 50 | **3** | 1 | 0 | 0 | 1 |
| 4 | $F_i - C_i$ | | 0 | -4 | -2 | 0 | 0 | 0 |
| 1 | $A_3$ | 0 | 46/3 | 0 | 5/3 | 1 | 0 | -1/3 |
| 2 | $A_4$ | 0 | 10/3 | 0 | **2/3** | 0 | 1 | -1/3 |
| 3 | $A_1$ | 4 | 50/3 | 1 | 1/3 | 0 | 0 | 1/3 |
| 4 | $F_i - C_i$ | | 200/3 | 0 | -2/3 | 0 | 0 | 4/3 |
| 1 | $A_3$ | 0 | 7 | 0 | 0 | 1 | -5/2 | 1/2 |
| 2 | $A_2$ | 2 | 5 | 0 | 1 | 0 | 3/2 | 1/2 |
| 3 | $A_1$ | 4 | 15 | 1 | 0 | 0 | -1/2 | 1/2 |
| 4 | $F_i - C_i$ | | 70 | 0 | 0 | 0 | 1 | 3 |

Симплекс-таблица составляется так:

В графе Базис записываются векторы переменных, принимаемые за базисные. На первом этапе это – $A_3$, $A_4$, $A_5$. Базисными будут переменные, каждая из которых входит только в одно уравнение системы, и нет такого уравнения, в которое не входила бы хотя бы одна из базисных переменных.

В следующий столбец $C_б$ записываются коэффициенты целевой функции, соответствующие каждой переменной. Столбец $B$ – столбец свободных членов. Далее идут столбцы коэффициентов $A_i$ при $i$ –й переменной.

Под столбцом свободных членов записывается начальная оценка $F_0 = \overline{C_б}\,\overline{B} = 0 \cdot 32 + 0 \cdot 20 + 0 \cdot 50 = 0$

Остальные оценки записываются под столбцами соответствующих векторов $\overline{A_i}$ .

$$F_1 - C_1 = \overline{C_б}\,\overline{A_1} - C_1 = 0\cdot 1 + 0\cdot 1 + 0\cdot 3 - 4 = -4$$

$$F_2 - C_2 = \overline{C_б}\,\overline{A_2} - C_2 = 0\cdot 2 + 0\cdot 1 + 0\cdot 1 - 2 = -2$$

Следует отметить, что оценки для базисных векторов всегда равны нулю.

Преобразование симплекс-таблицы ведется следующим образом:

Шаг 1: Проверяется критерий оптимальности, суть которого состоит в том, что все оценки $F_i - C_i$ должны быть неотрицательны. В нашем случае этот критерий не выполнен, поэтому переходим ко второму шагу.

Шаг 2: Для отрицательных оценок вычисляются величины:

$$\theta_1 = \min\left\{\frac{B_i}{A_i}\right\} = \min\left\{\frac{32}{1};\frac{20}{1};\frac{50}{3}\right\} = \frac{50}{3}$$

$$\theta_2 = \min\left\{\frac{32}{2};\frac{20}{1};\frac{50}{1}\right\} = 16$$

$$\theta_1\cdot(F_1 - C_1) = \frac{50}{3}\cdot(-4) = -\frac{200}{3}$$

$$\theta_2\cdot(F_2 - C_2) = 16\cdot(-2) = -32$$

Из этих элементов выбирается тот, для которого вычисленное произведение минимально, в нашем случае $-\frac{200}{3}$ минимально, поэтому в качестве так называемого разрешающего элемента выбирается третий элемент первого столбца – 3 (выделен в таблице).

Шаг 3: Третья строка таблицы делится на 3 и вычитается из первой и второй строк. В сущности, применяется метод исключения неизвестных (метод Жордана – Гаусса).

Таким образом, новыми базисными переменными становятся $A_3$, $A_4$, $A_1$.

Возвращаемся к шагу 1 и повторяем весь процесс.

Под столбцом свободных членов записывается начальная оценка

$$F_0 = \overline{C_б}\,\overline{B} = 0\cdot\frac{46}{3} + 0\cdot\frac{10}{3} + 4\cdot\frac{50}{3} = \frac{200}{3}$$

Остальные оценки записываются под столбцами соответствующих векторов $\overline{A_i}$ .

$$F_2 - C_2 = \overline{C_б}\,\overline{A_2} - C_2 = 0\cdot\frac{5}{3} + 0\cdot\frac{2}{3} + 4\cdot\frac{1}{3} - 2 = -\frac{2}{3}$$

$$F_5 - C_5 = \overline{C_б}\,\overline{A_5} - C_5 = 0\cdot\left(-\frac{1}{3}\right) + 0\cdot\left(-\frac{1}{3}\right) + 4\cdot\frac{1}{3} - 0 = \frac{4}{3}$$

Следует отметить, что оценки для базисных векторов всегда равны нулю.

Опять проверяется критерий оптимальности. Отрицательная оценка только одна – в столбце $A_2$.

$$\theta_2 = \min\left\{\frac{46}{5};5;50\right\} = 5$$

Вычисляем:

Разрешающим элементом будет второй элемент второго столбца – 2/3.

Новыми базисными переменными становятся $A_3$, $A_2$, $A_1$.

Делим вторую строку на 2 и вычитаем из третьей.

Умножаем вторую строку на 5/2 и вычитаем из первой.

$$F_0 = \overline{C_б}\,\overline{B} = 0\cdot 7 + 2\cdot 5 + 4\cdot 15 = 70$$

$$F_1 - C_1 = \overline{C_б}\,\overline{A_1} - C_1 = 0\cdot 0 + 2\cdot 0 + 1\cdot 4 - 4 = 0$$

$$F_2 - C_2 = \overline{C_б}\,\overline{A_2} - C_2 = 0\cdot 0 + 2\cdot 1 + 0\cdot 4 - 2 = 0$$

$$F_3 - C_3 = \overline{C_6}\,\overline{A_3} - C_3 = 0\cdot 1 + 0\cdot 2 + 0\cdot 4 - 0 = 0$$

$$F_4 - C_4 = \overline{C_6}\,\overline{A_4} - C_4 = 0\cdot\left(-\frac{5}{2}\right) + 2\cdot\frac{3}{2} + 4\cdot\left(-\frac{1}{2}\right) - 0 = 1$$

На этот раз отрицательных оценок нет,

$F_5 - C_5 = \overline{C_6}\,\overline{A_5} - C_5 = 0\cdot\left(\frac{1}{2}\right) + 2\cdot\frac{1}{2} + 4\cdot\left(\frac{1}{2}\right) - 0 = 3$ т.е. критерий оптимальности выполнен.

Таким образом, получается искомое значение целевой функции F(15; 5; 7; 0; 0) = 70, т. е. возвращаясь к системе неравенств, получаем:

$$F(15;5) = 60 + 10 = 70$$

Ответы, полученные различными методами, совпадают.

## Задания:

***Задание А. Привести к канонической форме следующие задачи ЛП***

**1.**

$2x_1 - x_2 + 3x_3 \le 5$

$x_1 + 2x_3 = 8$

$x_1 - 2x_2 \ge `1$

$x_j \ge o,\ x_2 \ge o,\ x_3 \ge o$

$F = x_1 - x_2 + 3x_3 \to min$

**2.**

$x_1 - 2x_2 + x_3 \ge 4$

$x_1 + x_2 - 3x_3 \le 9$

$x_1 + 3x_2 + 2x_3 = 10$

$x_j \ge o,\ x_2 \ge o,\ x_3 \ge o$

$F = 2x_1 + x_2 - x_3 \to max$

**3.**

$x_1 + 2x_2 - x_3 - 2x_4 + x_5 = 5$

$-2x_2 + 4x_3 + 4x_4 \le 4$

$x_2 \ge o,\ x_3 \ge o,\ X_5 \ge o$

$F = 2x_1 - x_2 + 3x_3 + x_4 - 2x_5 \to min$

**4.**

$-3x_1 + x_2 + 4x_3 - 2x_4 \ge 6$

$X_1 - 2x_2 + 3x_3 + x_4 + x_5 = 2$

$X_1 \ge o,\ x_3 \ge o,\ X_4 \ge o,\ X_5 \ge o$

$F = x_1 + 2x_2 + 3x_3 + 2x_4 + x_5 \to max$

**5.**

$2x_1 - x_2 + 6x_3 \le 12$

$3x_1 + 5x_2 - 12x_3 = 14$

$-3x_1 + 6x_2 + 4x_3 \le 18$

$x_j \ge o,\ x_2 \ge o,\ x_3 \ge o$

$F = 2x_1 - x_2 + x_3 \to min$

**6.**

$-4x_1 + 2x_2 + 5x_3 \le 12$

$6x_1 - 3x_2 + 4x_3 = 18$

$3x_1 + 3x_2 - 2x_3 \ge 16$

$x_j \ge o,\ x_2 \ge o,\ x_3 \ge o$

$F = -2x_1 + x_2 + 5x_3 \to max$

**Напишите задачи 1,2,5,6 в стандартных формах.**

***Задание Б. Привести к канонической форме следующие задачи ЛП***

$$1)\ \ x_1 - x_2 \ge 1;\ 2)\ x_1 - 2x_2 \le 1;\ 3)\ x_1 + x_2 \le 3;$$
$$x_j \ge 0,\ x_2 \ge 0$$
$$F = x_1 + 2x_2 \to max$$

Введем дополнительные переменные $x_3, x_4, x_5$. Причем в первое неравенство введем неотрицательную переменную $x_3$ со знаком минус, а во второе и в третье – со знаком плюс переменные $x_4, x_5$ запишем задачу в виде:

$$x_1 - x_2 - x_3 = 1;\ x_1 - 2x_2 + x_4 = 1;\ x_1 + x_2 + x_5 = 3$$
$$x_j \ge 0,\ j = 1\ldots5$$

Переведем *max* на *min*, домножив целевую функцию на (-1)

$$F = -x_1 - 2x_2 + 0x_3 + 0x_4 + 0x_5 \to min,$$

что и дает эквивалентную задачу в канонической форме.

### 1.4 Контрольные вопросы к защите занятия

1.1. Понятие математического программирования.
1.2. Формулировки задачи линейного программирования.
1.3. Понятие допустимого и оптимального плана.

### 1.5 Список рекомендуемой литературы

1. *Гладких Б.А.* Методы оптимизации и исследование операций для бакалавров информатики. Ч. I. Введение в исследование операций. Линейное программирование: Учебное пособие. – Томск: Изд-во НТЛ, 2009. – 200 с.

2. *Таха Х.А.* Введение в исследование операций. – 7-е изд.: Пер. с англ. — М.: Изд. дом «Вильямс», 2005. – 912 с.

3. *Гасс С.* Линейное программирование (методы и приложения): Пер. с англ. — М.: ГИФМЛ, 1964.

4. *Данциг Дж.* Линейное программирование, его применения и обобщения: Пер. с англ. — М.: Прогресс, 1966.

5. . *А. Схрейвер*. Теория линейного и целочисленного программирования: в 2-х томах.; перевод с английского. 1991г. 360 с.

6. *В.Г.Карманов.* Математическое программирование: Учебное пособие – 5-е издание, стереотип-М:ФИЗМАТ, 2001г.-264 с.

## 2. ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 7. ЦЕЛОЧИСЛЕННОЕ ПРОГРАММИРОВАНИЕ

### 2.1 Цель практического занятия:

изучение особенностей целочисленного программирования.

### 2.2 Задачи по теме занятия:

закрепить знания по разделам:

- понятие целочисленного программирования;
- метод Гомори;
- метод Ветвей и Границ;
- задачи, приводимые к целочисленным.

### 2.3 Основные теоретические сведения

**Определение.** *Целочисленное программирование – раздел математического программирования, где целевая функция линейна, а на множество, на котором ищется экстремум целевой функции, накладываются условия целочисленности* (например оптимизация выпуска изделий или их транспортировка).

**1. Метод Гомори (последовательных отсечений).** При решении многих задач (планирование мелкосерийного производства, распределение сообщений по маршрутам, выработка суждений типа «да–нет» и т.п.) нецелочисленное решение не имеет смысла. Попытка тривиального округления до целых значений приводит либо к нарушению ограничений задачи, либо к недоиспользованию ресурсов. Не трудно убедиться, что для произвольной линейной программы гарантировать целочисленность решения невозможно (за исключением программ типа классической транспортной задачи, где коэффициенты матрицы ограничений равны 1 или 0).

В случае двухмерной задачи проблема решается относительно просто путем выявления всех целочисленных точек, близких к границе множества планов, построения «выпуклой целочисленной оболочки» (выпуклого множества планов, содержащего все целочисленные планы) и решения задачи над этим множеством. В общем случае выдвигается *идея последовательного отсечения нецелочисленных оптимальных планов*:

симплексным методом отыскивается оптимальный план и, если он нецелочисленный, строится дополнительное ограничение, отсекающее найденный оптимальный план, но не отсекающее ни одного целочисленного плана.

Эта идея, принадлежащая Д. Данцигу и Р. Гомори, впервые была представлена в форме дополнительного ограничения: сумма небазисных компонент оптимального плана должна быть отлична от нуля (хотя бы одна из небазисных компонент должна быть ненулевой). В самом деле, оптимальный план с нулевыми значениями небазисных компонент этому условию не удовлетворяет, что подтверждает отсечение этого плана от исходного множества. Например, при

$$\sum_{j \notin Баз} X_j \geq 1 \qquad (1)$$

ограничение имеет вид:

$$f_k = \sum_{j \notin Баз} f_{kj} X_j - S^* , \quad S^* \geq 0 \quad , \qquad (2)$$

где $f_k$ – дробная часть компоненты плана (правой части ограничения) и $f_{kj}$ – дробная часть коэффициента при $X_j$ (целая часть числа наибольшее целое, не превышающее это – число; дробная часть числа равна разности между числом и его целой частью), $S^*$ – новая дополнительная переменная.

Объем преобразований можно уменьшить, если руководствоваться правилами:

1) выбирать в качестве базового для построения дополнительного ограничения уравнение, определяющее компоненту плана с наибольшей дробной частью;

2) для ввода в базис опорного плана расширенной задачи выбирать переменную, для которой достигается минимум из отношений абсолютных значений □j к значениям $f_{kj}$;

3) если одна из ранее введенных дополнительных переменных вошла в базис, ее и соответствующее ей уравнение можно отбросить (эта ситуация связана с появлением более жесткого условия, перекрывающего действие ранее введенного).

Появление дополнительного ограничения и дополнительной переменной вновь приводит к проблеме выбора начального опорного плана расширенной задачи и к использованию с этой целью искусственной переменной. Следует заметить, что если при поиске переменной, исключаемой из базиса, значение (≥) (определяемое с учетом дополнительного ограничения) соответствует этому ограничению, то можно отказаться от использования искусственной переменной (она все равно выведется из базиса на этом же шаге решения). Для целочисленных программ может обнаружиться отсутствие целочисленных планов (противоречивость ограничений).

Для предложенного метода доказана конечность процесса отсечений, но число этих отсечений непредсказуемо (может обнаружиться быстрое решение задач с десятками переменных и ограничений и весьма длительное для задач небольших размеров).

**Пример.** Для геометрической интерпретации процесса отсечений рассмотрим двухмерную задачу максимизации величины $L(X) = 3X_1 + X_2$

при условиях:

$$2X_1 + 3X_2 \geq 6$$
$$2X_1 - 3X_2 \leq 3$$
$$X_1 \geq 0,\ X_2 \geq 0 \text{ - целые.}$$

Вводим ослабляющие переменные

$$X_3 = 6 - [2X_1 + 3X_2] \geq 0$$
$$X_4 = 3 - [2X_1 - 3X_2] \geq 0$$

Решаем задачу до получения оптимального плана:

| C баз | Баз 1 | План X | 3 $X_1$ | 1 $X_2$ | 0 $X_3$ | 0 $X_4$ |
|---|---|---|---|---|---|---|
| 0 | $X_3$ | 6 | 2 | 3 | 1 | 0 |
| 0 | $X_4$ | 3 | **2** | -3 | 0 | 1 |
| $Z_j$ | | 0 | 0 | 0 | 0 | 0 |
| $\Delta_j$ | | | **-3** | -1 | 0 | 0 |

| C баз | Баз 2 | План X | 3 $X_1$ | 1 $X_2$ | 0 $X_3$ | 0 $X_4$ |
|---|---|---|---|---|---|---|
| 0 | $X_3$ | 3 | 0 | **6** | 1 | -1 |
| 3 | $X_1$ | 3/2 | 1 | -1.5 | 0 | 0.5 |
| $Z_j$ | | 4.5 | 3 | -4.5 | 0 | 1.5 |
| $\Delta_j$ | | | 0 | **-5.5** | 0 | 1.5 |

Получен оптимальный нецелочисленный план $X = (2.25, 0.5)$ с $L(X) = 7.25$.

Берем первое уравнение (согласно наибольшей дробной части компоненты плана) и строим дополнительное ограничение: $\frac{1}{2} = \frac{1}{6} X_3 + \frac{5}{6} X_4 - S_1 \ , \ S_1 \geq 0.$

Определяем вектор, подлежащий вводу в базис опорного плана расширенной задачи, согласно минимуму отношений $\Delta_j$ к значениям $f_{kj}$.

$\min \left( \frac{11/12}{1/6}, \frac{7/12}{5/6} \right)$

Так как соответствует $X_4$, то определяем, из какого уравнения выражать $X_4$ по обычному правилу:

$$\Theta = min \left( -, \frac{9/4}{1/4}, \frac{1/2}{5/6} \right) .$$

Поскольку этот минимум соответствует третьему (дополнительному) уравнению, то без использования искусственного базиса получаем оптимальный нецелочисленный план расширенной задачи $X = (2.1, 0.6)$.

По первому уравнению строим очередное дополнительное ограничение:

$$\frac{3}{5} = \frac{1}{5} X_3 + \frac{4}{5} S_1 - S_2 \ , \ S_2 \geq 0.$$

Из $\min \left( \frac{4/5}{1/5}, \frac{7/10}{4/5} \right)$ определяем, что вводу в базис новой задачи (с четырьмя уравнениями) подлежит переменная $S_1$. Так как $\Theta = min \left( -, \frac{2.1}{0.3}, -, \frac{3/5}{4/5} \right)$ соответствует четвертому (очередному дополнительному) уравнению, то опять-таки без использования искусственного базиса получаем оптимальный нецелочисленный план $X = (15/8, 3/4)$.

| C баз | Баз 5 | План X | 3 $X_1$ | 1 $X_2$ | 0 $X_3$ | 0 $X_4$ | 0 $S_1$ | 0 $S_2$ |
|---|---|---|---|---|---|---|---|---|
| 1 | $X_2$ | 3/4 | 0 | 1 | 1/4 | 0 | 0 | -1/4 |
| 3 | $X_1$ | 15/8 | 1 | 0 | 1/8 | 0 | 0 | 3/8 |
| 0 | $X_4$ | 3/2 | 0 | 0 | 1/2 | 1 | 0 | -3/2 |
| 0 | $S_1$ | 3/4 | 0 | 0 | 1/4 | 0 | 1 | -5/4 |
| $Z_j$ | | **51/8** | 3 | 1 | 5/8 | 0 | 0 | 7/8 |
| $\Delta_j$ | | | 0 | 0 | 5/8 | 0 | 0 | **7/8** |

По второму уравнению строим очередное дополнительное ограничение:

$$\frac{7}{8} = \frac{1}{8} X_3 + \frac{3}{8} S_2 - S_3 \ , \ S_3 \geq 0.$$

Из минимума отношений $\Delta_j$ к значениям $f_{kj}$ определяем, что вводу в базис новой задачи (с четырьмя уравнениями) подлежит переменная $S_2$. Так как выбор $\Delta$ определяется дополнительным уравнением, то опять-таки без использования искусственного базиса получаем оптимальный нецелочисленный план $X = (1, 4/3)$:

| C баз | Баз 6 | План X | 3 | 1 | 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|---|
| | | | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $S_1$ | $S_2$ | $S_3$ |
| 1 | $X_2$ | 4/3 | 0 | 1 | 1/3 | 0 | 0 | 0 | -2/3 |
| 3 | $X_1$ | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | $X_4$ | 5 | 0 | 0 | 1 | 0 | 0 | 0 | -4 |
| 0 | $S_1$ | 11/3 | 0 | 0 | 2/3 | 1 | 1 | 0 | -10/3 |
| 0 | $S_2$ | 7/3 | 0 | 0 | 1/3 | 0 | 0 | 1 | -8/3 |
| | $Z_j$ | **13/3** | 3 | 1 | 1/3 | 0 | 0 | 0 | 7/3 |
| | $\Delta_j$ | | 0 | 0 | **1/3** | 0 | 0 | 0 | 7/3 |

По четвертому уравнению строим новое ограничение:

$$\frac{2}{3} = \frac{2}{3} X_3 + \frac{2}{3} S_3 - S_4 \ , \ S_4 \geq 0 \ ,$$

видим необходимость ввода в базис переменной $X_3$ на основе последнего уравнения и получаем оптимальный целочисленный план $X$=(1,1):

| C баз | Баз 7 | План X | 3 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $S_1$ | $S_2$ | $S_3$ | $S_4$ |
| 1 | $X_2$ | 1 | 0 | 1 | 0 | 0 | 0 | 0 | -1 | 1/2 |
| 3 | $X_1$ | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | $X_4$ | 4 | 0 | 0 | 0 | 1 | 0 | 0 | -5 | 3/2 |
| 0 | $S_1$ | 3 | 0 | 0 | 0 | 0 | 1 | 0 | -4 | 1 |
| 0 | $S_2$ | 2 | 0 | 0 | 0 | 0 | 0 | 1 | -3 | 1/2 |
| 0 | $X_3$ | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | -3/2 |
| | $Z_j$ | 4 | 3 | 1 | 0 | 0 | 0 | 0 | 2 | 1/2 |
| | $\Delta_j$ | | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 1/2 |

Заметим, что объем последних симплексных таблиц можно было уменьшить за счет удаления строк и столбцов для дополнительных переменных, вошедших в базис.

$$f_k = \sum_{j \notin 3} R_{kj} X_j - S^* \ , \ S^* \geq 0 \ , \qquad (3)$$

Проиллюстрируем выполненный процесс в графической форме, выразив предварительно все переменные через $X_1$ и $X_2$:

$$/1/ \qquad S_1 = -\frac{1}{2} + \frac{1}{6} X_3 + \frac{5}{6} X_4 = 3 - 2X_1 - 2X_2 \geq 0 \ ;$$

$$/2/ \qquad S_2 = -\frac{3}{5} + \frac{1}{5} X_3 + \frac{4}{5} S_1 = 3 - 2X_1 + X_2 \geq 0 \ ;$$

$$/3/ \qquad S_3 = -\frac{7}{8} + \frac{1}{8} X_3 + \frac{3}{8} S_2 = 1 - X_1 \geq 0 \ ;$$

$$/4/ \qquad S_4 = -\frac{2}{3} + \frac{2}{3} X_3 + \frac{2}{3} S_3 = 4 - 2X_1 - 2X_2 \geq 0 \ .$$

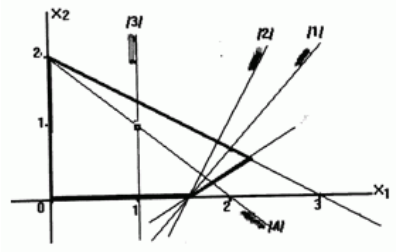



Рисунок 1

Рисунок 1 демонстрирует, что каждое очередное дополнительное ограничение отсекает от множества планов окрестность очередного нецелочисленного оптимального плана. Более того, видно, что после возникновения условия $S_3 \geq 0$ влияние предыдущих дополнительных ограничений отсутствует.

Ряд задач относят к категории частично целочисленных программ, если в них предъявляется требование целочисленности лишь к некоторому множеству переменных. Например, решая задачу минимизации линейной формы $2X_1 + 2X_2 + 5X_3$ при условиях:

$$3X_1 + 2X_2 + X_3 \geq 5$$
$$-X_1 + X_2 + 4X_3 \geq 7$$
$$X_1 - X_2 + 2X_3 \leq 4$$
$$X_1, X_2, X_3 \geq 0$$
$$X_1, X_2 \text{ - целые,}$$

приходим к оптимальному плану, определяемому таблицей вида

| C баз | Баз 3 | План X | 3 $X_1$ | 1 $X_2$ | 0 $X_3$ | 0 $X_4$ |
|---|---|---|---|---|---|---|
| 1 | $X_2$ | 0.5 | 0 | 1 | 1/6 | -1/6 |
| 3 | $X_1$ | 2.25 | 1 | 0 | 1/4 | 1/4 |
| $Z_j$ | | 7.25 | 3 | 1 | 11/12 | 7/12 |
| $\Delta_j$ | | | 0 | 0 | 11/12 | 7/12 |

| C баз | Баз 4 | План X | 2 $X_1$ | 2 $X_2$ | 5 $X_3$ | 0 $X_4$ | 0 $X_5$ | 0 $X_6$ |
|---|---|---|---|---|---|---|---|---|
| 2 | $X_2$ | 13/30 | 0 | 1 | 0 | -1/5 | -1/6 | 13/30 |
| 5 | $X_3$ | 55/30 | 0 | 0 | 1 | 0 | -1/6 | -1/6 |
| 2 | $X_1$ | 23/30 | 1 | 0 | 0 | -1/5 | 1/6 | -7/30 |
| $Z_j$ | | 7.9 | 2 | 2 | 5 | -4/5 | -5/6 | -13/30 |
| $\Delta_j$ | | | 0 | 0 | 0 | -4/5 | -5/6 | -13/30 |

Так как наибольшую дробную часть имеет (среди целочисленных по условиям задачи) компонента $X_1$, то выбираем третье уравнение и строим дополнительное ограничение в виде

$$\frac{23}{30} = \frac{23/30}{1 - 23/30}\left(-\frac{1}{5}\right) X_4 + \frac{1}{6} X_5 - \frac{23/30}{1 - 23/30}\left(-\frac{7}{30}\right) X_6 - S_1, \quad S_1 \geq 0,$$

т. е. $$\frac{23}{30} = \frac{23}{30} X_4 + \frac{1}{6} X_5 + \frac{23}{30} X_6 - S_1, \quad S_1 \geq 0.$$

**2. Метод ветвей и границ.** Как и в методе Гомори, решение начинается с поиска оптимального плана без учета целочисленности. Например, в рассмотренном примере решается двухмерная задача максимизации величины $L(X) = 3X_1 + X_2$ при условиях:

$$2X_1 + 3X_2 \geq 6$$
$$2X_1 - 3X_2 \leq 3$$
$$X_1 \geq 0, X_2 \geq 0.$$

Если компонента $X_k$ найденного плана равна нецелочисленной величине $T$, то строятся две расширенные задачи с дополнительными ограничениями $X_k \leq [T]$ и $X_k \geq [T] + 1$ соответственно (квадратные скобки определяют целую часть).

| $C$ баз | Баз 4(2) | План $X$ | 3 | 1 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|
| | | | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ |
| 0 | $X_4$ | 3 | 0 | 0 | -1 | 1 | -6 |
| 3 | $X_1$ | 1.5 | 1 | 1 | 1/2 | 0 | 3/2 |
| 1 | $X_2$ | 1 | 0 | 1 | 0 | 0 | -1 |
| | $Z_j$ | 5.5 | 3 | 1 | 3/2 | 0 | 7/2 |
| | $\Delta_j$ | | 0 | 0 | 3/2 | 0 | 7/2 |

Получив оптимальный план $X$ = (2.25, 0.5) со значением $L(X)$=7.25, построим две задачи. Первая из них получается из исходной, добавлением ограничения $X_2 \geq 0$, или с учетом неотрицательности $X_2$=0; вторая получается добавлением условия $X_2 \geq 1$. Решение первой из этих задач дает нецелочисленный оптимальный план $X = (1.5, 0)$ с $L(X)=4.5$.

Решение второй задачи (после серии симплексных преобразований) дает нецелочисленный оптимальный план $X$= (1.5,1) с несколько большим значением $L(X)$ = 5.5. значением $L(X)$= 5.5. Соответственно, берем эту ветвь и строим две подзадачи с условиями $X_1 \geq 1$и $X_1 \leq 2$. При решении второй из них обнаруживается противоречивость ограничений; первая же дает оптимальный план $X$ =(1,1).

Идея метода проста, но быстрота полученного решения является кажущейся. Во многих задачах метод ветвей и границ может потребовать существенной глубины просмотра и дает решение с гораздо большими затратами, чем метод Гомори. Тем не менее существует много задач, эффективно решаемых методом ветвей и границ.

### 2.4 Контрольные вопросы к защите занятия

1.4. Области допустимых значений векторов различных параметров.
1.5. Формулировка задачи непрерывной оптимизации в конечномерном пространстве.
1.6. Линейные задачи оптимизации.

### 2.5 Список рекомендуемой литературы

1. *А. Схрейвер* Теория линейного и целочисленного программирования: в 2-х томах.; перевод с английского. 1991г. 360с.

2. *Т. Ху* Целочисленное программирование и потоки в сетях; перевод с англ.. 1974г.

3. *В.Г. Карманов* Математическое программирование: Учебное пособие – 5-е издание, стереотип-М: ФИЗМАТ, 2001г.-264с.

4. *Е.Г. Белоусов* Введение в выпуклый анализ и целочисленное программирование. М.: Издательство МГУ, 1977г.

## 3. ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 8. НЕЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

### 3.1 Цель практического занятия:

знакомство с особенностями задач нелинейного (выпуклого) программирования.

### 3.2 Задачи по теме занятия:

закрепить знания по разделам:

- задачи условной оптимизации с ограничениями типа неравенств;
- теоремы Куна-Таккера.

### 3.3 Основные теоретические сведения

**Определение.** *Нелинейное программирование – раздел математического программирования, в котором целевая функция и ограничения рассматриваются как нелинейные зависимости.* В том числе: выпуклое программирование, где целевая функция выпукла и выпукло множество, на котором решается экстремальная задача. Частным

случаем является квадратичное программирование, где целевая функция квадратична, а ограничениями являются линейные равенства и неравенства. В многоэкстремальных задачах обычно выделяются специализированные классы задач, часто встречающихся в приложениях, например, задачи о минимизации на выпуклом множестве вогнутых функций.

**Задача условной оптимизации с ограничениями типа неравенств.** Пример:

$$\min_{\mathbf{X}\in D} \Phi(\mathbf{X})=\Phi(\mathbf{X}^*)=\Phi^*, \qquad (1)$$

где $\Phi(\mathbf{X})$ произвольная функция, $D=\{\mathbf{X}|g(\mathbf{X})\geq 0\}=\{\mathbf{X}|g_i(\mathbf{X})\geq 0, i\in[1,m]\}\subset R^n$ - непустое, ограниченное замкнутое множество.

Далее используются понятия множителей Лагранжа и функции Лагранжа для задачи нелинейного программирования с ограничениями типа неравенств. Функция Лагранжа для задачи (1) с ограничениями (2) определяется формулой

$$L(\mathbf{X},\lambda)=\Phi(\mathbf{X})+\sum_{i=1}^{m}\lambda_i g_i(\mathbf{X})=\Phi(\mathbf{X})+\lambda^T g(\mathbf{X}), \qquad (2)$$

где $\lambda=(\lambda_i, i\in[1,m])$ - $(m*1)$– вектор множителей Лагранжа**.**

Вводятся понятия активных и неактивных ограничений. В точке локального минимума задачи (1), (2) каждое из ограничений $g_i(\mathbf{X}^*)\geq 0, i\in[1,m]$ выполняется либо в виде равенства $g_i(\mathbf{X}^*)=0$, либо в виде неравенства $g_i(\mathbf{X}^*)>0$. Ограничения первого вида называются *активными ограничениями*. Остальные ограничения называются *неактивными ограничениями*.

Вводится понятие условия регулярности для задачи нелинейного программирования с ограничениями типа неравенств. Если точка $\mathbf{X}^*\in D$ и ограничения $g_{i_j}(\mathbf{X}^*)\geq 0, i_j\in[1,s], s\leq m$ активны, то условие линейной независимости векторов называется условием регулярности ограничивающих функций $\nabla g_{i_j}(\mathbf{X}^*)\geq 0, i_j\in[1,s]$ в точке $\mathbf{X}^*$. Это условие означает, что, например, при $n$ =2 количество ограничивающих функций, проходящих через точку $\mathbf{X}^*$, не должно превышать 2 и в точке $\mathbf{X}^*$ векторы $\nabla g_1(\mathbf{X})$, $\nabla g_2(\mathbf{X})$ не должны быть коллениарны Например, на рис. 1 в ситуации (а) количество ограничивающих функций, проходящих через точку $\mathbf{X}^*$, превышает размерность вектора варьируемых параметров, в ситуации (б) в точке $\mathbf{X}^*$ градиенты $\nabla g_1(\mathbf{X})$, $\nabla g_2(\mathbf{X})$ ограничивающих функций коллениарны.

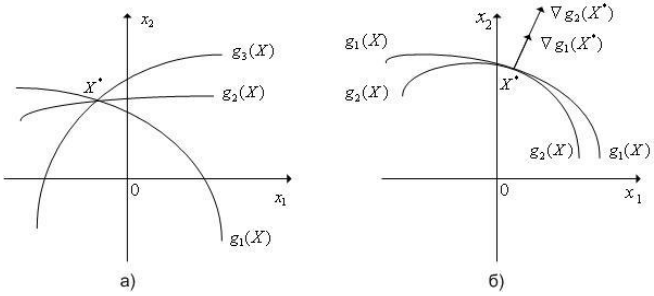



Рис. 1. Ситуации, в которых не выполняется условие регулярности двумерной задачи

Большое значение в теории и практике решения задач нелинейного программирования имеет теорема Куна-Таккера для задачи условной оптимизации с ограничениями типа неравенств.

**Теорема Куна-Таккера**. *Если функция Ф(X) и функции* $g_i(\mathbf{X})\geq 0, i\in[1,m]$ *имеют непрерывные частные производные в некоторой окрестности точки* $\mathbf{X}^*$ *и эта точка является точкой локального минимума функции Ф(X) при ограничениях* $g_i(\mathbf{X}^*)\geq 0$, *удовлетворяющих в точке* $\mathbf{X}^*$ *условию регулярности ограничивающих функций, тогда существуют такие неотрицательные множители Лагранжа* $\lambda_i, i\in[1,m]$, *что для функции Лагранжа* $L(\mathbf{X},\lambda)$ *точка* $\mathbf{X}^*$ *является стационарной точкой функции, т.е.*

$$\nabla_X L(\mathbf{X}^*,\lambda)=\nabla\Phi(\mathbf{X}^*)+\sum_{i=1}^{m}\lambda_i\nabla g_i(\mathbf{X}^*)=0\bullet \qquad (3)$$

В отличие от правила множителей Лагранжа, теорема Куна-Таккера требует знакоопределенности множителей Лагранжа $\lambda_i, i\in[1,m]$. Следует отметить, что теорема не запрещает равенство нулю всех множителей Лагранжа $\lambda_i, i\in[1,m]$. Поясним смысл теоремы на примере.

**Пример**. Рассмотрим двумерную ($n$=2) задачу нелинейного программирования (1), (2), в которой область допустимых значений $D$ задается тремя ограничивающими функциями, т.е. $D=\{\mathbf{X}|g(\mathbf{X})\geq 0\}=\{\mathbf{X}|g_i(\mathbf{X})\geq 0, i\in[1,3]\}\subset R^2$. Положим, что множество $D$ имеет вид, представленный на рисунке 2.

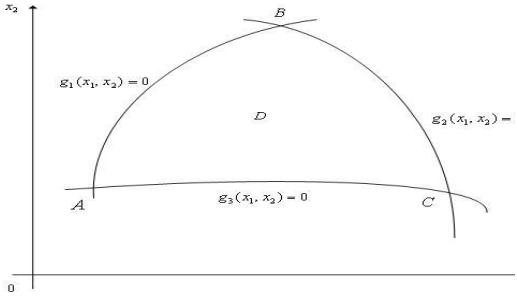



Рис. 2. Вид множества $D$

Для всех граничных точек области $D$, очевидно, выполняются условия регулярности ограничивающих функций. Если точка $\mathbf{X}^*$ находится внутри множества $D$ (т. е. является стационарной точкой функции *Ф(X)* ), то теорема будет справедлива, если положить все множители Лагранжа $\lambda_i, i\in[1,m]$ равными нулю.

Пусть точка $\mathbf{X}^*$ находится на одной из дуг, например, на дуге АВ, т.е. пусть ограничение $g_1(\mathbf{X})\geq 0$ является активным ограничением, а остальные ограничения – неактивными ограничениями. Тогда в этой точке $g_1(\mathbf{X}^*)=0$ и справедливость теоремы вытекает из правила множителей Лагранжа для задачи с ограничениями типа равенств, если положить $\lambda_2=\lambda_3=0$.

Пусть точка $\mathbf{X}^*$ находится в одной из угловых точек множества $D$, например, в точке $B$, т. е. пусть ограничения $g_1(\mathbf{X})\geq 0$, $g_2(\mathbf{X})\geq 0$ являются активными ограничениями, а ограничение $g_3(\mathbf{X})\geq 0$ - неактивным ограничением. Тогда можно положить $\lambda_3=0$ и справедливость теоремы вытекает из правила множителей Лагранжа для задачи с ограничениями типа равенств.

Теорема Куна-Таккера показывает, что в ее условиях вместо задачи условной оптимизации (1), (2) можно решать задачу безусловной оптимизации $\min_{\mathbf{X}\in R^n}\left(\Phi(\mathbf{X})+\sum_{i=1}^{m}\lambda_i g_i(\mathbf{X})\right)$.

*Необходимым условием существования локального минимума* этой задачи в некоторой точке $\mathbf{X}^*\in R^n$ является условие $\nabla_x\left(\Phi(\mathbf{X}^*)+\sum_{i=1}^{m}\lambda_i g_i(\mathbf{X}^*)\right)=0$.

В условиях применимости теоремы Куна-Таккера существуют такие неотрицательные множители Лагранжа $\lambda_i, i\in[1,m]$, что имеют место следующие равенства:

$$\nabla_X L(\mathbf{X}^*,\lambda)=0, \tag{4}$$

$$\nabla_X L(\mathbf{X}^*,\lambda)=g(\mathbf{X}^*)\geq 0\bullet \tag{5}$$

Здесь равенство (5) фактически повторяет равенство (4), и по условиям теоремы Куна-Таккера точка $\mathbf{X}^*$ удовлетворяет всем ограничениям, т.е. $g_i(\mathbf{X}^*)\geq 0, i\in[1,m]$.

***Теорема 2*** устанавливает условия, при выполнении которых точка Куна – Таккера автоматически соответствует оптимальному решению задачи нелинейного программирования.

*Достаточность условий Куна – Таккера* поясним на примере следующей задачи:

Минимизировать *F(X)*. При ограничениях

$g_i(x)\geq 0,\ j=1,\ 2,\ \ldots,\ J,$

$h_k(x)=0,\ k=1,\ 2,\ \ldots,\ K,\ x=(x_1,\ x_2,\ \ldots,\ x_N).$

Пусть целевая функция *F(X)* выпуклая, все ограничения в виде неравенств содержат вогнутые функции $G_j(X),\ J=1,\ \ldots,\ J$, а ограничения в виде равенств содержат линейные функции $H_k(X)$, удовлетворяющие условиям Куна – Таккера

$$\Delta f(x)-\sum_{j=1}^{J}u_j\cdot\Delta g_j(x)-\sum_{k=1}^{K}v_k\cdot\Delta h_k(x)=0,$$

$$\begin{cases} g_j(x)\geq 0, & j=1,\ 2,\ \ldots,\ J,\\ h_k(x)=0, & k=1,\ 2,\ \ldots,\ K,\\ u_j g_j(x)=0, & j=1,\ 2,\ \ldots,\ J,\\ u_j\geq 0, & j=1,\ 2,\ \ldots,\ J, \end{cases}$$

То x* – оптимальное решение задачи нелинейного программирования.

*Замечание.* Если условия теоремы 2 выполняются, то нахождение точки Куна – Таккера обеспечивает получение оптимального решения задачи нелинейного программирования. Теорему можно использовать для доказательства оптимальности данного решения.

*Пример.* Минимизировать $f(x)=x_1^2-x_2$

При ограничениях $h_1(x)=x_1+x_2-6=0,\ g_1(x)=x_1-1\geq 0,\ g_2(x)=26-x_1^2-x_2^2\geq 0.$

*Решение.* С помощью теоремы 2 докажем, что решение $x_1^*=1,\ x_2^*=5$ является оптимальным. Имеем

$$\Delta f(x)=(2x_1,-1),\quad H_f(x)=\begin{bmatrix}2 & 0\\ 0 & 0\end{bmatrix}.$$

Так как матрица $H_f(X)$ положительно полуопределена при всех *X*, функция *F(X)* оказывается выпуклой. Первое ограничение в виде неравенства содержит линейную функцию $G_1(X)$, которая одновременно является как выпуклой, так и вогнутой. Для того, чтобы показать, что функция $G_2(X)$ является вогнутой, вычислим

$$\Delta g_2(x)=(-2x_1,\ -2x_2),\quad H_{g_2}(x)=\begin{bmatrix}-2 & 0\\ 0 & -2\end{bmatrix}.$$

Поскольку матрица $H_{g_2}(x)$ отрицательно определена, функция $G_2(X)$ является вогнутой. Функция $H_1(X)$ входит в линейное ограничение в виде равенства. Следовательно, все условия Теоремы 2 выполнены; если покажем, что $X^* = (1, 5)$ – точка Куна – Таккера, то действительно установим оптимальность решения $X^*$.

Условия Куна – Таккера для данного примера имеют вид

$$2x_1 - u_1 + 2x_1u_2 - v_1 = 0,$$
$$-1 + 2x_2u_2 - v_1 = 0,$$
$$x_1 - 1 \geq 0,$$
$$26 - x_1^2 - x_2^2 \geq 0,$$
$$x_1 + x_2 - 6 = 0,$$
$$u_1(x_1 - 1) = 0,$$
$$u_2(26 - x_1^2 - x_2^2) = 0,$$
$$u_1, u_2 \geq 0.$$

Точка $X^* = (1,5)$ удовлетворяет полученным ограничениям и, следовательно, является допустимой. Полученные уравнения принимают вид:

$$2 - u_1 + 2u_2 - v_1 = 0,$$
$$-1 + 10u_2 - v_1 = 0.$$

Положив $V_1 = 0$, получим $U_2 = 0.1$ и $U_1 = 2.2$. Таким образом, решение $X^* = (1, 5)$, $U^* = (2.2, 0.1)$ и $V_1 = 0$ удовлетворяет условиям Куна – Таккера. Поскольку условия *Теоремы 2* выполнены, то $X^* = (1, 5)$ – оптимальное решение задачи. Заметим, что существуют также и другие значения $U_1, U_2, V_1$, которые удовлетворяют условиям Куна – Таккера, построенным для задачи.

### 3.4 Контрольные вопросы к защите занятия

1.Теорема Куна-Таккера для задачи условной оптимизации с ограничениями типа неравенств.

2. Понятие о множителе Лагранжа.

3. Необходимое и достаточное условия локального минимума

### 3.5 Список рекомендуемой литературы

*1. Гольштейн Е. Г.* Выпуклое программирование. Элементы теории. М., 1970;

*2. Зангвилл У. И.* Нелинейное программирование. Единый подход. М., 1973.

## 4. ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 9. СТОХАСТИЧЕСКОЕ ПРОГРАММИРОВАНИЕ

### 4.1 Цель практического занятия:

знакомство с методами стохастического программирования.

### 4.2 Задачи по теме:

закрепить знания по разделам:

- характеристики видов неопределённости;
- понятие вероятностного пространства;
- выбор программы действий в условиях неопределённости.

### 4.3 Основные теоретические сведения

**Определение.** *Стохастическое программирование – решение задач оптимизации в условиях неопределённости, когда точные значения результатов принимаемых решений неизвестны, и используются методы программирования с применением понятий теории вероятности.*

В общем случае неопределённость принято подразделять на стохастическую (имеется информация о распределении вероятности на множестве результатов), поведенческую (имеется информация о влиянии на результаты поведения участников), природную (имеется информация только о возможных результатах и отсутствует о связи между решениями и результатами) и априорную (нет информации и о возможных результатах).

Задача обоснования решений в условиях неопределённости всех типов, кроме априорной, сводится к сужению исходного множества альтернатив на основе информации, которой располагает лицо, принимающее решение (ЛПР). Качество рекомендаций для принятия решений в условиях стохастической неопределённости повышается при учёте таких характеристик личности ЛПР, как отношение к своим выигрышам и проигрышам, склонность к риску. Обоснование решений в условиях априорной неопределённости возможно построением алгоритмов адаптивного управления

Для строгой формулировки задач стохастического программирования используется понятие вероятностного пространства – это тройка $(\Omega, \mathfrak{F}, \mathsf{P})$, где:

- $\Omega$ – произвольное множество, элементы которого называются элементарными событиями, исходами или точками;
- $\mathfrak{F}$ – сигма-алгебра подмножеств $\Omega$, называемых (случайными) событиями. Си́гма-а́лгебра – это алгебра множеств, замкнутая относительно операции счётного объединения.
- $\mathsf{P}$ – вероятностная мера или вероятность, т.е. сигма-аддитивная конечная мера, такая что $\mathsf{P}(\Omega) = 1$.

Элементарные события (элементы $\Omega$), по определению, - это исходы случайного эксперимента, из которых в эксперименте происходит ровно один. Каждое случайное событие (элемент $\mathfrak{F}$) – это подмножество $\Omega$. Говорят, что в результате эксперимента произошло случайное событие $A \subseteq \Omega$, если (элементарный) исход эксперимента является элементом $A$.

Требование, что $\mathfrak{F}$ является сигма-алгеброй подмножеств $\Omega$, позволяет, в частности, говорить о вероятности случайного события, являющегося объединением счетного числа случайных событий, а также о вероятности дополнения любого события.

Простым и часто используемым примером вероятностного пространства является конечное пространство. Пусть $\Omega$ – конечное множество, содержащее $|\Omega| = n$ элементов. В качестве сигма-алгебры удобно взять семейство подмножеств $\Omega$. Его часто символически обозначают $2^{\Omega}$. Общее число членов этого семейства, т.е. число различных случайных событий, как раз равно $2^{|\Omega|}$, что объясняет обозначение.

Выбор программы действий в условиях неопределённости предполагает ориентацию на «ожидаемую ценность» результатов таких действий. Для её оценки часто используется термин «математическое ожидание». В силу важности этого понятия дадим его определение и рассмотрим некоторые характеристики. Пусть задано вероятностное пространство $(\Omega, \mathfrak{F}, \mathbb{P})$ и определённая на нём случайная величина X. То есть, по определению, $X : \Omega \to \mathbb{R}$ – измеримая функция. Если существует интеграл Лебега от $X$ по пространству $\Omega$, то он называется математическим ожиданием, или средним (ожидаемым) значением, и обозначается *M[X]* или E*[X]*.

$$M[X] = \int_{\Omega} X(\omega)\, \mathbb{P}(d\omega).$$

Если имеется функция распределения случайной величины $F_X(X)$, то её математическое ожидание задаётся интегралом Лебега — Стилтьеса:

$$M[X] = \int_{-\infty}^{\infty} x\, dF_X(x); x \in \mathbb{R}.$$

Если имеется дискретная случайная величина *X,* имеющая распределение $\mathbb{P}(X = x_i) = p_i, \sum_{i=1}^{\infty} p_i = 1$, то из определения интеграла Лебега следует, что $M[X] = \sum_{i=1}^{\infty} x_i p_i$.

Если $X$ – положительная целочисленная случайная величина (частный случай дискретной), имеющая распределение вероятностей $\mathbb{P}(X=j)=p_j,\ j=0,1,...;\quad \sum_{j=0}^{\infty} p_j = 1$, то её математическое ожидание может быть выражено через производящую функцию последовательности $\{p_i\}$ $P(s)=\sum_{k=0}^{\infty} p_k s^k$ как значение первой производной в единице: M[X]=P1(1). Если математическое ожидание $X$ бесконечно, то $\lim_{s\to 1} P'(s)=\infty$ и будем писать $P'(1)=M[X]=\infty$

Возьмем производящую функцию *Q(S)* последовательности «хвостов» распределения $\{q_k\}$ $q_k=\mathbb{P}(X>k)=\sum_{j=k+1}^{\infty} p_j;\quad Q(s)=\sum_{k=0}^{\infty} q_k s^k.$

Эта производящая функция связана с определённой ранее функцией *P(S)* свойством: $Q(s)=\frac{1-P(s)}{1-s}$ при $|s|<1$. Из этого по теореме о среднем следует, что математическое ожидание равно просто значению этой функции в единице: *M[X]=P1(1)=Q(1).*

Например, пусть случайная величина имеет дискретное равномерное распределение, то есть $\mathbb{P}(X=x_i)=\frac{1}{n},\ i=1,\dots,n.$. Тогда её математическое ожидание $M[X]=\frac{1}{n}\sum_{i=1}^{n} x_i$ е равно среднему арифметическому всех принимаемых значений.

Если случайная величина имеет непрерывное равномерное распределение на интервале $[a,b]$, где $a<b$. Тогда её плотность имеет вид $f_X(x)=\frac{1}{b-a}\mathbf{1}_{[a,b]}(x)$ и математическое ожидание $M[X]=\int_a^b \frac{x}{b-a}\,dx=\frac{a+b}{2}$.

В практических расчетах вероятность определяется в соответствии со смыслом задачи. Часто нет причин считать, что один элементарный исход предпочтительнее другого. Тогда вероятность определяется: $\mathrm{P}(A)=\frac{n_A}{n}$, где $A\subset\Omega$, и $|A|=n_A$- число элементарных исходов, принадлежащих $A$.

В частности, вероятность любого элементарного события: $\mathrm{P}(\{\omega\})=\frac{1}{n},\ \forall\omega\in\Omega.$

*Пример. Эксперимент с бросанием уравновешенной монеты.* Берутся два события: выпадение герба ( Г ) и выпадение решки ( Р ), то есть $\Omega=\{\Gamma,\mathrm{P}\}$. Тогда $\mathfrak{F}=\{\{\Gamma\},\{\mathrm{P}\},\{\Gamma,\mathrm{P}\},\varnothing\}$ и вероятность можно посчитать:

$$\mathrm{P}(\{\Gamma\})=\frac{1}{2},\ \mathrm{P}(\{\mathrm{P}\})=\frac{1}{2},\ \mathrm{P}(\{\Gamma,\mathrm{P}\})=1,\ \mathrm{P}(\varnothing)=0.$$

Таким образом, определена тройка $(\Omega,\mathfrak{F},\mathrm{P})$ – вероятностное пространство, в рамках которого можно рассматривать различные задачи.

*Задачи с участием случайного процесса различаются по трем признакам.*

1. По характеру решений (детерминированный или случайный вектор, чистые или смешанные стратегии).

2. По выбору показателя качества, т. е. целевой функции. Если находится**:**

а) математическое ожидание от целевой функции, т.е. $M[F]=M[CX]\to\max$, то такие задачи называются *M-моделями*;

б) если минимизируется дисперсия целевой функции $M\left[\left(CX-\overline{CX}\right)^2\right]\to\min$, то такие задачи называются *V-моделями;*

в) если определяется вероятность превышения целевой функцией некоторого порога $F_0$, т. е. $P[F\ge F_0]=P[CX\ge F_0]$, то такие задачи называются *P-моделями.*

3. По способу расчленения ограничений задачи могут быть:

а) С *построчными* вероятностными ограничениями, когда учитываются стохастические связи только в одной строке: $P\left\{\sum_j^n a_{ij}x_j\ge b_j\right\}\ge\alpha_i, i=\overline{1,m},$

где $a_i$ – вероятность соблюдения условий,

$a_{0i} = 1 - a_i$ ¾ вероятность несоблюдения условий. (от 0 до 1)

б) *С общими* вероятностными ограничениями $P[AX \geq B] \geq a$ , когда могут быть коррелированы параметры не только строки, но и между строками. В такой модели не учитывается сравнительная важность отдельных ограничений.

### 4.4 Контрольные вопросы к защите занятия

1. Характеристика вероятностного пространства;
2. Характеристика видов неопределённости;
3. Характеристика задач с участием случайного процесса.

### 4.5 Список рекомендуемой литературы

1. Агальцов В.П., Волдайская И.В. Математические методы в программировании: Учебник – М . : ИД «ФОРУМ» : ИНФРА-М, 2006. – 224с.

2. Партика Т.Л., Попов И.И. Математические методы: Учебник /М.: ФОРУМ: ИНФРА, 2005.

3. Интернет сайт: http://ru.wikipedia.org/wiki/.

4. Костевич Л. Математическое программирование / Изд. «Новое знание», 2003.

## 5. ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 10. ДИНАМИЧЕСКОЕ ПРОГРАММИРОВАНИЕ

### 5.1 Цель практического занятия:

знакомство с методами динамического программирования.

### 5.2 Задачи по теме:

закрепить знания по разделам:

- идея динамического программирования;
- понятие о рекурсивной последовательности простых подзадач;
- восходящее и нисходящее динамическое программирование.

### 5.3 Основные теоретические сведения

**Определение.** *Динамическое программирование – способ решения сложных задач путём разбиения их на более простые подзадачи; применим к задачам, представимых как набор перекрывающихся подзадач, сложность которых чуть меньше исходной.* В этом случае время вычислений, по сравнению с «наивными» методами, можно значительно сократить.

Ключевая идея. чтобы решить поставленную задачу, требуется решить её отдельные части задачи (подзадачи), после чего объединить решения подзадач в одно общее решение. Часто многие из этих подзадач одинаковы. Подход динамического программирования состоит в том, чтобы решить каждую подзадачу только один раз, сократив тем самым количество вычислений. Это особенно полезно в случаях, когда число повторяющихся подзадач экспоненциально велико.

Метод *динамического программирования сверху* основан на простом запоминании результатов решения тех подзадач, которые могут повторно встретиться в дальнейшем. *Динамическое программирование снизу* включает в себя переформулирование сложной задачи в виде рекурсивной последовательности более простых подзадач.

Оптимальность подструктуры в динамическом программировании означает, что оптимальное решение подзадач меньшего размера может быть использовано для решения исходной задачи. Например, кратчайший путь в графе из вершины (*s*) в вершину (*t*) может быть найден так: сначала считается кратчайший путь до *t* из всех вершин, смежных с *s*, а затем, учитывая веса ребер, которыми *s* соединена со смежными вершинами, выбирается

лучший путь до *t*. В общем случае мы можем решить задачу, в которой присутствует оптимальная подструктура, проделывая следующие три шага:

- разбиение задачи на подзадачи меньшего размера;
- нахождение оптимального решения подзадач рекурсивно, проделывая такой же трехшаговый алгоритм;
- использование полученного решения подзадач для конструирования решения исходной задачи.

*Перекрывающиеся подзадачи* в динамическом программировании означают подзадачи, которые используются для решения некоторого количества задач (не одной) большего размера (то есть несколько раз проделывается одно и то же). Например, вычисление последовательности Фибоначчи, $F_3 = F_2 + F_1$ и $F_4 = F_3 + F_2$ даже в таком тривиальном случае вычисления всего двух чисел Фибоначчи мы уже посчитали $F_2$дважды. Если продолжать дальше и посчитать $F_5$, то $F_2$ посчитается ещё два раза, так как для вычисления $F_5$ будут нужны опять $F_3$ и $F_4$. Получается следующее: простой рекурсивный подход будет расходовать время на вычисление решение для задач, которые он уже решал. Чтобы избежать такого хода событий требуется сохранять решения подзадач, которые уже решали, и, когда снова потребуется решение подзадачи, вместо того, чтобы вычислять его заново, оно просто достается из памяти. Этот подход называется кэширование.

В целом динамическое программирование пользуется следующими свойствами задачи: перекрывающиеся подзадачи; оптимальная подструктура; возможность запоминания решения часто встречающихся подзадач. Языки программирования могут запоминать результат вызова функции с определенным набором аргументов, чтобы ускорить «вычисление по имени». В некоторых языках такая возможность встроена (например, *Scheme, Common Lisp, Perl*), а в некоторых требует дополнительных расширений (C++). К задачам динамического программирования можно отнести:

- задача о наибольшей общей подпоследовательности: даны две последовательности, требуется найти самую длинную общую подпоследовательность;
- задача поиска наибольшей увеличивающейся подпоследовательности: дана последовательность, требуется найти самую длинную возрастающую подпоследовательность;
- задача о редакционном расстоянии (расстояние Левенштейна): даны две строки, требуется найти минимальное количество стираний, замен и добавлений символов, преобразующих одну строку в другую;
- задача о вычислении чисел Фибоначчи;
- задача о порядке перемножения матриц: даны матрицы $A_1$, ..., $A_n$, требуется минимизировать количество скалярных операций для их перемножения;
- задача о рюкзаке: из неограниченного множества предметов со свойствами «стоимость» и «вес», требуется отобрать некое число предметов таким образом, чтобы получить максимальную суммарную стоимость при ограниченном суммарном весе;
- алгоритм Флойда-Уоршелла: найти кратчайшие расстояния между всеми вершинами взвешенного ориентированного графа;
- алгоритм Беллмана – Форда: найти кратчайший путь во взвешенном графе между двумя заданными вершинами, а также максимальное независимое множество вершин в дереве: дано дерево, найти максимальное множество вершин, никакие две из которых не связаны ребром.

Например, классической задачей на последовательности является следующая. Последовательность Фибоначчи $F_n$ задается формулами: $F_1 = 1$, $F_2 = 1$, $F_n = F_{n-1} + F_{n-2}$ при $n > 1$. Необходимо найти $F_n$ по номеру $n$. Один из способов решения, который может показаться логичным и эффективным, – решение с помощью рекурсии:

```
int F(int n) {
  if (n < 2) return 1;
  else return F(n - 1) + F(n - 2);
}
```

Используя такую функцию, будем решать задачу «с конца» – будем шаг за шагом уменьшать $n$, пока не дойдем до известных значений. Как можно заметить, такая, казалось бы, простая программа уже при $n = 40$ работает заметно долго. Это связано с тем, что одни и те же промежуточные данные вычисляются по несколько раз – число операций нарастает с той же скоростью, с какой растут числа Фибоначчи – экспоненциально.

Один из выходов из данной ситуации – сохранение уже найденных промежуточных результатов с целью их повторного использования:

```
int F(int n) {
 if (A[n] != -1) return A[n];
 if (n < 2) return 1;
 else {
  A[n] = F(n - 1) + F(n - 2);
  return A[n];
 }
}
```

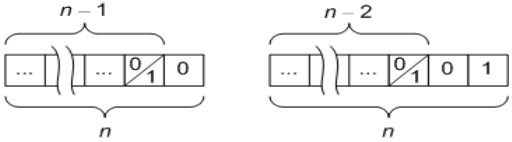
Приведенное решение является корректным и эффективным. Но для данной задачи применимо и более простое решение:

```
F[0] = 1;
F[1] = 1;
for (i = 2; i < n; i++) F[i] = F[i - 1] + F[i - 2];
```

Такое решение можно назвать решением «с начала» – первым делом заполняем известные значения, затем находим первое неизвестное значение ($F_3$), потом следующее и т.д., пока не дойдем до нужного. Именно такое решение и является классическим для динамического программирования: мы сначала решили все подзадачи (нашли все $F_i$ для $i<n$), затем, зная решения подзадач, нашли ответ ($F_n = F_{n-1} + F_{n-2}$, $F_{n-1}$ и $F_{n-2}$ уже найдены).

*Пример.* Решение задач одномерного динамического программирования позволит понять суть схемы расчета оптимального решения. Пусть задача заключается в нахождении некоторого числа $T$ при исходных данных $n_1, n_2, ..., n_k$, т.е. можно говорить о функции $T(n_1, n_2, ..., n_k)$, значение которой является искомым ответом. Тогда подзадачами будем считать задачи $T(i_1, i_2, ..., i_k)$ при $i_1< n_1, i_2< n_2, ..., i_k< n_k$. Предлагаемая задача встречается в различных вариациях. Например, найти число последовательностей нулей и единиц длины $n$, в которых не встречаются две идущие подряд единицы. При $n< 32$ полный перебор потребует нескольких секунд, а при $n = 64$ полный перебор не осуществим в принципе. Для решения сведем исходную задачу к подзадачам. При $n = 1$, $n = 2$ ответ очевиден. Допустим, что мы уже нашли $K_{n-1}$, $K_{n-2}$ – число таких последовательностей длины $n – 1$ и $n –2$.

Посмотрим, какой может быть последовательность длины $n$. Если последний ее символ равен 0, то первые $n – 1$ – любая правильная последовательность длины $n – 1$ (не важно, заканчивается она нулем или единицей - следом идет 0). Таких последовательностей всего $Kn–1$. Если последний символ равен 1, то предпоследний символ обязательно должен быть равен 0 (иначе будет две единицы подряд), а первые $n–2$ символа - любая правильная последовательность длины $n–2$, число таких последовательностей равно $Kn – 2$.




Таким образом, $K_1 = 2$, $K_2 = 3$, $K_n = K_{n-1} + K_{n-2}$ при $n> 2$. То есть данная задача фактически сводится к нахождению чисел Фибоначчи.

Более сложными являются задачи двумерного динамического программирования. Например, задачи о маршрутах на прямоугольном поле, предполагающая, что в разных формулировках необходимо посчитать число маршрутов или найти маршрут, который является лучшим в некотором смысле. Для всех таких задач характерным является то, что каждый отдельный маршрут не может пройти два или более раз по одной и той же клетке.

**Примеры.** *Задача 1.* Дано прямоугольное поле размером *n*m* клеток. Можно совершать шаги длиной в одну клетку вправо или вниз. Посчитать, сколькими способами можно попасть из левой верхней клетки в правую нижнюю.

*Решение.* В некоторую клетку с координатами (*i*,*j*) можно прийти только сверху или слева, то есть из клеток с координатами (*i – 1, j*) и (*i, j – 1*):Таким образом, для клетки (*i, j*) число маршрутов *A[i][j]* будет равно

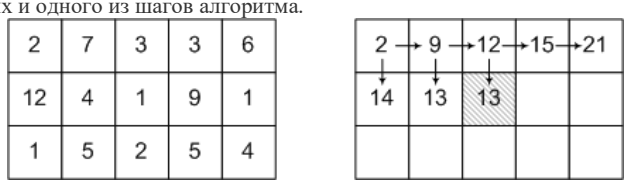
$$A[i-1][j] + A[i][j-1], \qquad (1)$$

то есть задача сводится к двум подзадачам. В данной реализации используется два параметра (*i* и *j*) поэтому здесь мы говорим о двумерном динамическом программировании. Теперь можно пройти последовательно по строкам (или по столбцам) массива A, находя число маршрутов для текущей клетки согласно формуле (1), предварительно в A[0][0 поместив число 1.

*Задача 2***.** Дано прямоугольное поле размером *n*m* клеток. Можно совершать шаги длиной в одну клетку вправо, вниз или по диагонали вправо-вниз. В каждой клетке записано некоторое натуральное число. Необходимо попасть из верхней левой клетки в правую нижнюю. Вес маршрута вычисляется как сумма чисел со всех посещенных клеток. Необходимо найти маршрут с минимальным весом. В клетку с координатами (*i, j*) можно попасть из клеток с координатами *(i – 1, j), (i, j – 1)* и *(i – 1, j – 1).*

Допустим, что для каждой из этих трех клеток уже найден маршрут минимального веса, а сами веса поместили в *W[i–1][j],W[i][j – 1], W[i – 1][j – 1]*. Чтобы найти минимальный вес для (*i, j*), необходимо выбрать минимальный из весов *W[i – 1][j], W[i][j – 1], W[i – 1][j – 1]* и прибавить к нему число, записанное в текущей клетке: *W[i][j] = min(W[i–1][j], W[i][j – 1], W[i – 1][j – 1]) + A[i][j];*

Данная задача осложнена тем, что необходимо найти не только минимальный вес, но и сам маршрут. Поэтому в другой массив дополнительно для каждой клетки будем записывать, с какой стороны в нее надо попасть. На рисунке приведен пример исходных данных и одного из шагов алгоритма.



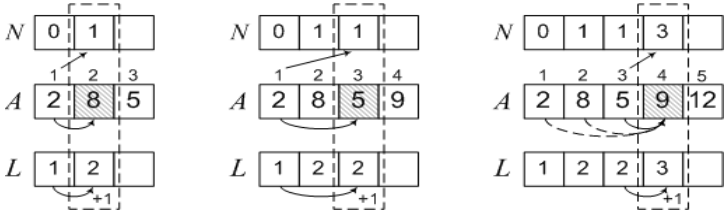

В каждую из уже пройденных клеток ведет ровно одна стрелка. Эта стрелка показывает, с какой стороны необходимо прийти в эту клетку, чтобы получить минимальный вес, записанный в клетке. После прохождения всего массива необходимо проследить сам маршрут из последней клетки, следуя по стрелкам в обратную сторону.

*Задача 3: «на подпоследовательности».* Дана последовательность целых чисел. Необходимо найти ее самую длинную строго возрастающую подпоследовательность.

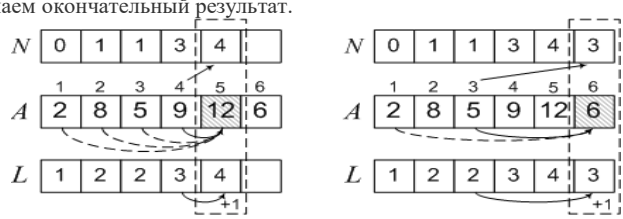
Начнем решать задачу с начала - будем искать ответ, начиная с первых членов данной последовательности. Для каждого номера *i* будем искать наибольшую возрастающую подпоследовательность, оканчивающуюся элементом в позиции *i*. Пусть исходная последовательность хранится в массиве *A*. В массиве *L* будем записывать длины максимальных подпоследовательностей, оканчивающихся текущим элементом. Пусть мы нашли все *L[i]* для $1<=i<=k-1$. Теперь можно найти *L[k]* следующим образом. Просматриваем все элементы *A[i]* для $1<= i < k-1$. Если $A[i]<A[k]$, то *k*-ый элемент может стать продолжением подпоследовательности, окончившейся элементом *A[i]*. Длина полученной подпоследовательности будет на 1 больше *L[i]*. Чтобы найти *L[k]*, необходимо перебрать все *i* от 1 до $k-1$: $L[k]=max(L[i]) + 1$, где максимум берется по всем *i* таким, что $A[i] < A[k]$ и $1 <= i < k$.

Здесь максимум из пустого множества будем считать равным 0. В этом случае текущий элемент станет единственным в выбранной последовательности, а не будет продолжением одной из предыдущих. После заполнения массива *L* длина наибольшей возрастающей подпоследовательности будет равна максимальному элементу *L*. Чтобы восстановить саму подпоследовательность, можно для каждого элемента также сохранять номер предыдущего выбранного элемента, например, в массив *N*.

Рассмотрим решение задачи на примере последовательности 2, 8, 5, 9, 12, 6. Поскольку до 2 нет ни одного элемента, то максимальная подпоследовательность содержит только один элемент - *L[1]* = 1, а перед ним нет ни одного *N[1]* = 0. Далее, 2 < 8, поэтому 8 может стать продолжением последовательности с предыдущим элементом. Тогда *L[2] = 2, N[2] = 1.*




Меньше *A*[3] = 5 только элемент *A*[1] = 2, поэтому 5 может стать продолжением только одной подпоследовательности, которая содержит 2. Тогда *L[3] = L[1] + 1 = 2, N[3]= 1*, так как 2 стоит в позиции с номером 1. Аналогично выполняем еще три шага алгоритма и получаем окончательный результат.




Теперь выбираем максимальный элемент в массиве *L* и по массиву *N* восстанавливаем подпоследовательность 2, 5, 9, 12.

## 5.4 Контрольные вопросы к защите занятия

1. Основной принцип динамического программирования Беллмана.
2. Идея оптимального управления методом динамического программирования.
3. Различие между восходящим и нисходящим методами динамического программирования.

## 5.5 Список рекомендуемой литературы

1. *Томас Х. Кормен и др.* Алгоритмы: построение и анализ = INTRODUCTION TO ALGORITHMS. 2-е изд. М.: «Вильямс», 2006. 1296 с.

2. *Беллман Р.* Динамическое программирование. М.: Изд. иностранной лит. 1960.

*3. Кормен, Т., Лейзерсон, Ч., Ривест, Р., Штайн, К.* Динамическое программирование // Алгоритмы: построение и анализ = Introduction to Algorithms / Под ред. И. В. Красикова. 2-е изд. М.: Вильямс, 2005. 1296 с.

*4. Акулич И.Л.* Математическое программирование в примерах и задачах. – М.: Высшая школа, 1986. 319 с.