1. **Предмет изучения, структура курса. Общее понятие задачи оптимизации. Классификация задач оптимизации.**

**Математическое программирование** –раздел высшей математики, посвященный решению задач оптимизации, т.е. задач, связанных с нахождением экстремумов функций нескольких переменных, при наличии ограничений на переменные.

Методами мп решаются задачи о распределении ресурсов, планировании выпуска продукции, ценообразования, транспортные задачи и т.д.

Линейное программирование, комбинаторные алгоритмы, динамическое программирование, транспортная задача, метод ветвей и границ, сетевое планирование, минимальные покрывающие деревья, оптимизационные алгоритмы на графах, потоки в сетях

Решение задачи математического программирования осуществляется в 4 этапа.

1. Построение математической модели.
2. Классификация задачи.
3. Выбор метода решения.
4. Вычисление.

Классификация: характеру взаимосвязи между переменными (линейные, нелинейные)

­– по характеру изменения переменных (непрерывные, дискретные)

* по учету временного фактора(статические, динамические)
* по наличию информации (полной определенности, неполной информации, неопределенности)
* по кол-ву критериев (простые-однокритериальные, сложные-многокритериальные)

Задача оптимизации - задача нахождения экстремума целевой функции в некоторой области пространства, ограниченной набором неравенств.

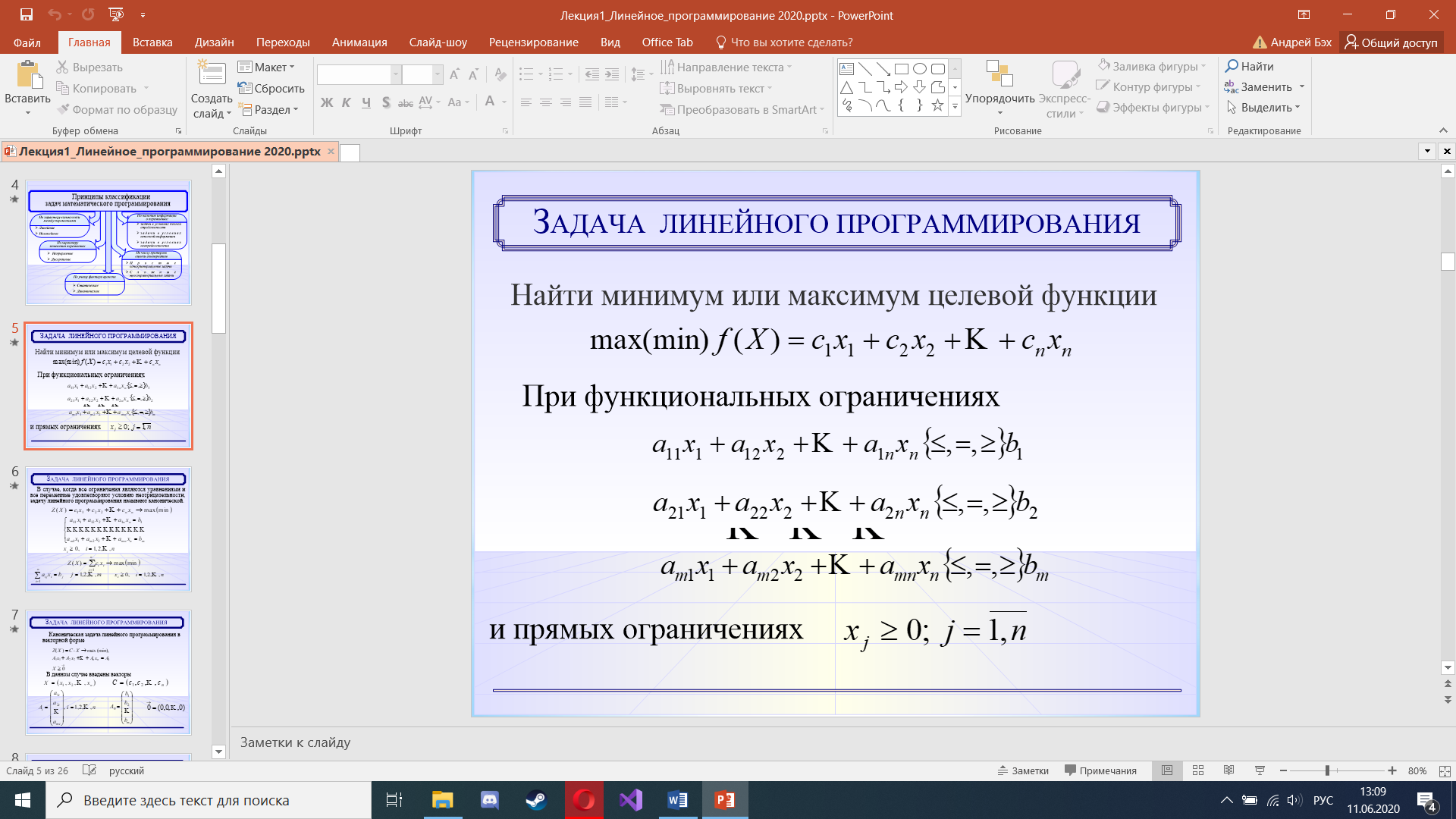
Методы: метод мат. программирования (нахождение экстремумов целевой функции при ограничениях), лин. Программирования (линейные зависимости между переменными), нелин.программирование (нелинейная зависимость между переменными).

1. **Общая формулировка задачи линейной оптимизации. Формы записи задач линейной оптимизации.**

Задача линейного программирования (ЛП) состоит в нахождении минимума (или максимума) линейной функции при линейных ограничениях.

Существует общая, каноническая и стандартная формы записи.

Общая форма:



В случае, когда все ограничения являются уравнениями и все переменные удовлетворяют условию неотрицательности, задачу лп называют канонической.

1. **Геометрический метод решения задачи линейной оптимизации.**

Графический или геометрический метод решения задачи лп состоит из 2 этапов:

1. Построение пространства допустимых решений, удовлетворяющих всем ограничениям модели.
2. Нахождение оптимального решения среди всех точек пространства допустимых решений.

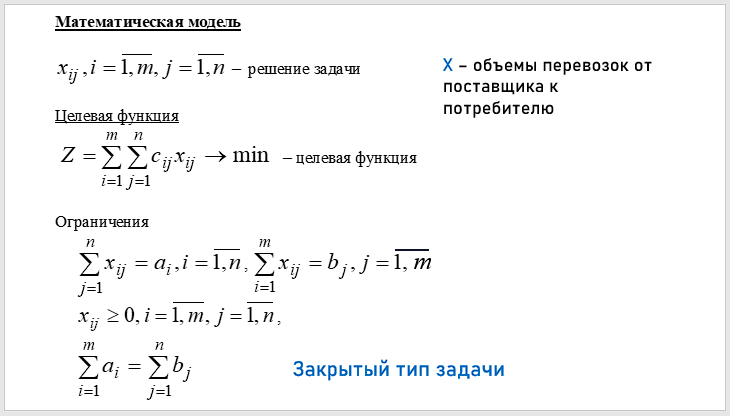
Алгоритм решения:

1. Строится область допустимых решений
2. Строится вектор нормали к линии уровня с точкой приложения в начале координат
3. Перпендикулярно вектору нормали проводится одна из линий уровня, проходящая через начало координат
4. Линии уровня перемещаются до положения опорной прямой. На этой прямой и будут находится мин или макс функции

В зависимости от вида области допустимых решений и целевой функции задача может иметь единственное решение, бесконечное множество решений или не иметь ни одного оптимального решения.

1. **Транспортная задача. Математическая модель транспортной задачи. Алгоритм решения транспортной задачи. Методы построения исходного опорного решения.**

Транспортные задачи – специальный класс задач линейного программирования. Назначение транспортной задачи – определение объемов перевозок из пункт\ов отправления в пункты назначения с минимальной суммарной стоимостью перевозок.



**Этапы:**

1. Построение начального базисного решения : метод северо-западного угла, метод наименьшей стоимости (минимального элемента), метод Фогеля.
2. Итеративный процесс поиска оптимального решения (метод потенциалов).

Общая транспортная задача с m пунктами отправления и n пунктами назначения имеет m+n ограничений в виде равенств, по одному на каждый пункт отправления и назначения. Т. к. транспортная задача д.б. сбалансированной, то одно из этих равенств избыточно. Т.о. транспортная задача имеет m+n+1 независимых ограничений, отсюда вытекает, что начальное базисное решение состоит из m+n+1 базисных переменных.

1. **Метод потенциалов нахождения оптимального решения транспортной задачи.**

Используется для итеративного процесса поиска оптимального решения. Каждой строке и каждому столбцу транспортной таблицы ставятся в соответствие числа(потенциалы) ui(поставщик) и vj(потребитель), которые удовлетворяют уравнению

ui + vj = cij

Потенциалы: u1, n, где n - кол-во поставщиков

v1, m, где m - кол-во потребителей

определяем потенциалы для всех переменных, присваивая случайной переменной значение, обычно u1 = 0

считаем потенциалы для свободных клеток (если все <= 0, то решение является оптимальным)

выбираем наибольший из положительных потенциалов и создаем цикл, в котором перемещаем минимальное значение покругу

для нового решения считаем значение целевой функции и потенциалы

1. **Общие принципы решения задач оптимизации методом ветвей и границ.**

Метод ветвей и границ – это общий алгоритмический метод решения задач комбинаторной оптимизации.

Метод является вариацией полного перебора с отсевом подмножеств допустимых решений, заведомо не содержащих оптимальных решений.

в основе метода лежат 2 процедуры: процедура ветвления(позволяет разбивать множества на непересекающиеся подмножества) и процедура вычисления верхней и нижней границ

Гамильтонов цикл - цикл, который проходит через каждую вершину графа ровно по 1 разу

Гамильтонов путь - путь, который проходит ерез каждую вершину графа ровно 1 раз.

Гамильтонов путь отличается от цикла тем, что у пути начальные и конечные точки могут не совпадать, в отличие от цикла.

Гамильтонов цикл является гамильтоновым путем.

Процедура вычисления нижней и верхней границы основывается на 2 утверждениях:

1) Изменение всех элементов строки матрицы расстояний на одно и то же число не влияет на выбор оптимального маршрута коммивояжера.

2) Изменение всех элементов столбца матрицы расстояний на одно и то же число не влияет на выбор оптимального маршрута коммивояжера.

Для начала находим минимальные элементы в каждой строке матрицы и проводим операцию приведения по строкам

Далее проводим точно такую же операцию по столбцам

Матрица приведенная по строкам и столбцам называется полностью приведенной

Приведение матрицы расстояний не влияет на выбор оптимального маршрута коммивояжера

нижняя граница длины колцевого кратчайшего маршрута будет равна сумме констант приведения

далее применяем процедуру ветвления:

Для начала надо найти дугу ветвления. Для этого надо для каждого элемента равного 0 посчитать сумму минимального элемента строки и столбца

Элемент с максимальной суммой будет элементом ветвления

Проверить включаем мы элемент или нет можно посчитав её нижнюю границу без включения и с включением, если при включении нижняя граница меньше, чем без включения, то включаем

Если мы включаем дугу в граф, то мы вычеркиваем элементы столбца и строки, в котором она стояла и ставим бесконечность в противоположный элемент

1. **Комбинаторные методы решения оптимизационных задач. Генерация подмножеств заданного множества.**

Пусть  – конечное множество мощности  **Множество всех подмножеств** множества  называется ***булеаном***  и обозначается  Количество элементов булеана множества  вычисляется по формуле

Алгоритм генерации множества всех подмножеств основывается на взаимно однозначном соответствии между элементами булеана  множества  и всеми целыми числами множества , записанными в двоичном виде

Построение элементов булеана множества  сводится к следующему алгоритму:

1. Пронумеровать элементы заданного множества  начиная с нуля.
2. Сформировать битовую последовательность  состоящую из  двоичных нулей. Пронумеровать элементы этой последовательности справа налево, начиная с нуля.
3. Последовательно выполнить шаги 4 и 5 алгоритма  раз.
4. Выбрать из множества  элементы с номерами  для которых  Полученное подмножество будет являться элементом булеана  В первом случае не будет выбран ни один элемент (пустое подмножество) множества  так как исходная последовательность  состоит только из нулей.
5. Интерпретируя битовую последовательность как целое положительное число, увеличить это число на единицу.
6. **Комбинаторные методы решения оптимизационных задач. Генерация сочетаний.**

Сочетания из n элементов по m, называют их соединения, каждое из которых содержит ровно m различных элементов и которые отличаются хотя бы одним элементом

Поэтому генерация множества  может быть сведена к генерации булеана и выбору из него всех подмножеств с мощностью 

****

1. **Комбинаторные методы решения оптимизационных задач. Генерация перестановок.**

Наиболее известным методом построения множества  всех перестановок конечного множества  является ***алгоритм Джонсона – Троттера***. Алгоритм подразумевает, что все элементы множества  можно единственным способом перечислить в порядке возрастания. (Pn=n!)

Построение множества всех перестановок с помощью алгоритма Джонсона – Троттера сводится к следующей процедуре:

1. Построить первую перестановку. Первая перестановка – это последовательность всех элементов множества  перечисленных в порядке возрастания. Стрелки всех элементов последовательности направлены влево.
2. Найти наибольший мобильный элемент в текущей перестановке. Если в последовательности нет мобильного элемента, то построены все перестановки элементов множества  – алгоритм закончил свою работу.

3. Поменять местами наибольший мобильный элемент и элемент, на который указывает стрелка наибольшего мобильного элемента.

4. Найти все элементы, большие, чем мобильный элемент (если они есть) и изменить их стрелки на противоположное направление.

5. Перейти к пункту 2.

1. **Комбинаторные методы решения оптимизационных задач. Генерация размещений.**

Множество размещений  из  по  можно получить, перестановкой элементов всех сочетаний множества . Другими словами, для получения множества размещений  требуется сначала сгенерировать все сочетания по  элементов из множества  а затем все перестановки элементов для каждого сочетания.

1. **Динамическое программирование. Вычислительная схема решения задачи динамического программирования (на примере решения задачи о рюкзаке).**

Задача динамического управления состоит в поиске оптимального управления, переводящего систему из начального состояния в конечное, и обеспечивающего экстремум целевой функции

аддитивная функция - сумма эффектов решений, принимаемых на отдельных этапах управляемого процесса

мультипликативная функция - произведение одношаговых функций

Принцип оптимальности белмана: каким бы ни было состояние системы перед очередным шагом, шаговое управление необходимо выбирать таким образом, чтобы выигрыш на данном шаге + оптимальный выигрыш на всех последующих шагах был максимальным

Алгорит реализации метода ДП:

1) определяются состояния системы и набор шаговых управлений

2) определяется выигрыш на данном шаге управления

3)определяется изменение состояния системы при управлении

4)проводится условная оптимизация последнего шага

5)проводится условная оптимизация предпоследнего шага и так до первого, вычисляя условный выигрыш

6)проводится безусловная оптимизация

Решение задачи о рюкзаке:

1) определение входных данных

2) создание таблицы

3) заполнение таблицы

4) оптимальное решение

5) восстановление решения

Дистанция левенштейна определяется между двумя строками и равна минимальному количеству операций вставки, удаления или изменения одного элемента, необходимых для превращения одной строки в другую

1. **Рекурсивные алгоритмы.**

***Рекурсивный алгоритм*** – это алгоритм, решающий задачу путем сведения ее к решению одной или нескольких таких же задач, но в сокращенном их варианте.

Рекурсивный алгоритм - алгоритм решающий задачу, путем сведения её к решению одной или нескольких таких же задач, но в сокращенном их варианте.

рекурсивная функция - функция, которая вызывает саму себя

Классические примеры рекурсивных функций являются функции для вычисления факториала, чисел фибоначчи

системный стек - специальная секция памяти, резервируемая операционной системой для хранения контекста

глубина рекурсии -количество вызовов функции в цепочке вызовов

разделяй и властвуй - рекурсивные функции, применяемые для решения задач, используют более 1 рекурсивного вызова, каждый из которых работает приблизительно с половиной входных данных

1. **Математические основы сетевого планирования. Основные понятия теории графов.**

Граф – это математическая модель, с помощью которой удобно представлять бинарное отношение.

Рассмотрим множество , которое будем назвать ***множеством*** ***вершин***, и набор  упорядоченных пар , называемый ***множеством дуг***. Пара  называется ***графом***. Запись  обозначает граф с именем , состоящий из множества вершин  и множества дуг .

Две соединенные дугой вершины графа, называются ***смежными вершинами***.

Считается, что дуга  ***выходит из вершины***  и ***входит в вершину*** . При этом говорят, что вершины  и  ***инцидентны*** дуге , а дуга  инцидентна вершинам  и .

Вершины  и  дуги  называют соответственно ***начальной*** и ***конечной*** вершинами этой дуги.

Дуги, которые выходят и входят в одну и ту же вершину, называются ***петлями***.

Вершины, не имеющие смежных, называются ***изолированными вершинами***.

Очевидно, что множество дуг  можно интерпретировать как бинарное отношение, а  – множество, на котором это бинарное отношение строится.

Если множество дуг  не является симметричным отношением, то такой граф называется ***ориентированным графом***.

Если множество дуг – симметричное отношение, то соответствующий граф называется ***неориентированным*** ***графом***.

***Существует три основных способа представления графов:***

* ***матрица смежности,***
* ***матрица инцидентности***
* ***списки смежных вершин.***





1. **Математические основы сетевого планирования. Кратчайшие пути между вершинами графа.**

Граф, дуги которого имеют вес, называется взвешенным. Вес – действительное число, присваиваемое дуге. Вес присваивается с помощью весовой функции.



Поиск кратчайшего пути между двумя вершинами графа является одной из часто используемых в приложениях задач. Наиболее известными способами решения этой задачи являются ***алгоритмы Дейкстры***, ***Беллмана*** *–* ***Форда*** и ***Флойда – Уоршолла***. Здесь будет рассмотрен только алгоритм Дейкстры

Алгоритм Дейкстры последовательно преобразовывает исходный граф  в дерево кратчайших путей . ***Дерево кратчайших путей*** – это граф , обладающий следующими свойствами:

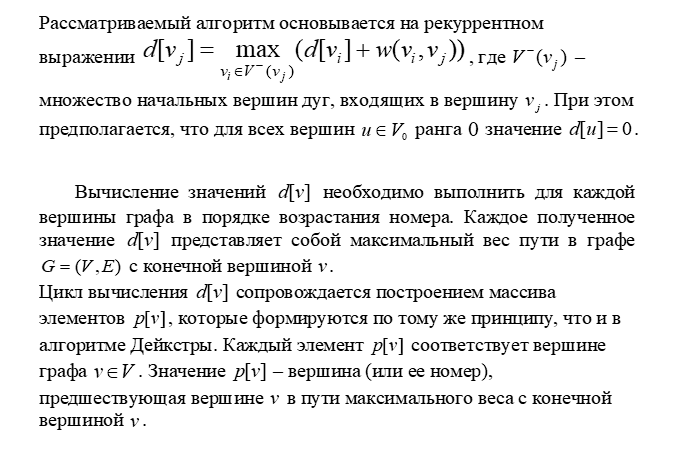
1. ;
2.  – ориентированное дерево с корнем ;
3.  – множество достижимых вершин графа  из ;
4. для каждой вершины  путь из  в  в дереве  является кратчайшим путем в графе .

Идея алгоритма состоит в следующем: поддерживается множество 𝑆, состоящее из вершин, для которых мы уже нашли кратчайшие пути (т.е. 𝑑[𝑢] = 𝛿(𝑠, 𝑢)). Далее алгоритм добавляет к 𝑆 вершину 𝑣 ∈ 𝑉 ∖𝑆 с наименьшим 𝑑[𝑣], а затем производит релаксацию всех рёбер, выходящих из 𝑣, после чего цикл повторяется

1. **Математические основы сетевого планирования. Максимальные пути между вершинами графа.**

***Максимальным путем  называется путь с максимальным весом ***. В общем случае, в одном графе может быть несколько максимальных путей.

Построение максимального пути ****** во взвешенном ориентированном графе  возможно, если в нем нет контуров с положительным весом. Если в графе есть такой контур, то некоторые пути могут иметь сколь угодно большой вес, т.к. каждый обход контура увеличивает вес пути на величину веса этого контура.



1. **Сетевые модели. Применение сетевых моделей. Сетевые графики.**

В практике управления сложными системами широко применяются методы сетевого планирования и управления (СПУ). Например: Автономные системы и робототехника, Игровая индустрия, Компьютерное зрение

Основные разновидности этих методов:

* ***Метод критического пути*** (СРМ) применяется тогда, когда операции, входящие в состав комплекса работ, имеют известные строго определенные продолжительности (являются ***детерминированными***).
* В свою очередь, ***метод оценки и обзора программ*** (РЕRТ) применяется при планировании проектов, для которых характерна ***неопределенность*** в оценке затрат времени, необходимого для выполнения отдельных операций.

СПУ включает три основных этапа:

* ***Структурное планирование***
* ***Календарное планирование***
* ***Оперативное управление***.

***Структурное планирование*** начинается с разбиения проекта на четко определенные операции. Затем строится ***сетевой график***, который представляет взаимосвязи работ проекта.

***Календарное планирование*** предусматривает построение календарного графика, определяющего моменты начала и окончания каждой работы и другие временные характеристики сетевого графика.

В ходе ***оперативного управления*** используются сетевой и календарный графики для составления периодических отчетов о ходе выполнения проекта.

*Сетевой моделью* называется модель, отражающая комплекс работ (операций) и событий, связанных с реализацией некоторого проекта в их логической и технологической последовательности и связи.

Сетевая модель обычно представляется ***сетевым графиком***, определяющим логическую взаимосвязь и взаимообусловленность входящих в него элементарных операций (работ).

Последовательность дуг или ребер, ведущая от некоторой вершины к другой, образует ***путь***.

Как правило, вершины, называемые ***событиями***, соответствуют моментам времени начала или окончания одной или нескольких операций, а дуги – операциям.

Различают три вида событий: ***исходное***, ***завершающее*** и ***промежуточное***. С ***исходного*** события начинается выполнение комплекса операций. ***Завершающее*** событие соответствует достижению конечной цели.

Сетевые графики с несколькими завершающими событиями называются ***многоцелевыми***. К ***промежуточным*** относятся все прочие события.

Различают три вида операций:

1. действительная операция ***(------>) требует затрат времени и ресурсов (разработка проекта, подвоз материалов, выполнение монтажных работ и т. п.);***
2. операция - ожидание ***(-.-.-.-.->) требует только затрат времени (затвердение бетона, сушка штукатурки перед началом малярных работ, рост растений и т. д.);***
3. фиктивная операция ***(- - - - - - - >) - технологическая или ресурсная зависимость в выполнении некоторых операций***

При построении сетевых графиков соблюдается ряд правил:

1. в сети не должно быть событий (кроме исходного), в которые не входит ни одна дуга;
2. не должно быть событий (кроме завершающего), из которых не выходит ни одной дуги;
3. сеть не должна содержать замкнутых контуров (циклов);
4. ***любая пара событий сетевого графика может быть соединена не более, чем одной дугой;***
5. ***номер*** начального ***события любой операции должен быть меньше номера ее*** конечного ***события.***
6. **Минимальные покрывающие деревья. Основные алгоритмы нахождения минимального остовного дерева.**

Связный подграф  графа  являющийся деревом и содержащий все его вершины, называется покрывающим деревом, или остовным деревом.

В общем случае один граф может иметь несколько покрывающих деревьев.

Покрывающее дерево  графа , имеющее минимальную сумму длин его ребер, называется минимальным покрывающим деревом, или минимальным остовным деревом.

В общем случае граф может иметь несколько минимальных покрывающих деревьев.

Наиболее известными алгоритмами построения минимального остовного дерева являются алгоритмы Крускала и Прима.

Построение минимального остовного дерева в этих алгоритмах осуществляется поэтапно. Начинается построение с пустого множества ребер  На каждом этапе множество  пополняется одним ребром, причем так, что множество ребер  всегда остается подмножеством ребер искомого минимального остовного дерева. Задача сводится к поиску необходимого ребра (обычно его называют безопасным ребром) на каждом шаге алгоритма.

Алгоритмы Крускала и Прима относятся к классу алгоритмов, называемых жадными алгоритмами. Такое название эти алгоритмы получили за стратегию, заключающуюся в принятии на каждом шаге локально оптимального решения (жадного решения), в предположении, что такая стратегия приведет к конечному оптимальному решению.

**Алгоритм Крускала** работает по тому же принципу. Отличие только в порядке объединения вершин в пошагово формирующемся минимальном покрывающем дереве.

Первоначально в алгоритме Крускала неориентированный связный граф  с заданной на его ребрах весовой функцией   разбивается на максимальное количество подграфов, каждый из которых является деревом. Очевидно, что каждое такое дерево будет представлять собой одну вершину графа, а общее количество таких подграфов будет 

Далее из множества  ребер графа  поочередно в порядке возрастания длины выбираются ребра. При этом возможны два случая:

1) концевые вершины лежат в разных подграфах разбиения;

2) обе концевые вершины лежат в одном подграфе разбиения.

В первом случае из двух подграфов, которые можно соединить выбранным ребром, образуется один общий, включающий все вершины и ребра этих подграфов, а также новое связующее ребро. Очевидно, что такое объединение не может образовать циклы, а следовательно, объединенный подграф тоже является деревом.

Во втором случае не выполняется никаких новых построений.

Из условия связности исходного графа  очевидно, что итогом работы такого алгоритма будет дерево, связывающее все вершины графа (т. е. остовное дерево).

Минимальность остовного дерева следует из того, что на каждом его шаге выбиралось безопасное ребро для совокупности ребер двух подграфов.

Построение минимального остовного дерева с помощью алгоритма **Прима** начинается с выбора произвольной вершины исходного неориентированного связного графа. Выбранная вершина окрашивается.

На последующих шагах просматриваются все окрашенные вершины и анализируются все ребра исходного графа, у которых одна концевая вершина окрашена, а другая нет. Среди всех таких ребер выбирается ребро с наименьшей длиной. Неокрашенная вершина этого ребра окрашивается, а само оно добавляется в формируемое минимальное остовное дерево.

Алгоритм заканчивает свою работу, когда все вершины графа станут окрашенными. Сформированное множество выбранных ребер будет составлять искомое минимальное остовное дерево.

1. **Оптимизационные алгоритмы на графах. Алгоритм поиска в ширину.**

Алгоритм подразумевает, что задана исходная (стартовой) вершина, и основывается на простом правиле: при выборе очередной вершины предпочтение отдается ближайшей. При этом считается, что все дуги графа имеют единичную длину. Сначала посещается стартовая вершина, затем все вершины, смежные ей (т. е. находящиеся на расстоянии 1), после чего вершины, находящиеся на расстоянии 2 от стартовой и т.д.

Очевидно, что алгоритм BFS посетит только те вершины, для которых существует последовательность дуг, связывающая с ними стартовую вершину. Обычно, по окончании работы алгоритма, осуществляется проверка на полноту обхода и, если имеются вершины, которые не посещались, то среди них выбирается любая, назначается стартовой и алгоритм выполняется снова.

1. **Оптимизационные алгоритмы на графах. Алгоритм поиска в глубину.**

Как и для поиска в ширину, задается стартовая вершина. Алгоритм описывается следующим образом: для каждой не пройденной вершины, начиная со стартовой, необходимо найти все смежные вершины и повторить поиск для каждой.

1. **Оптимизационные алгоритмы на графах. Топологическая сортировка.**

**Топологическая сортировка** − это процедура упорядочивания вершин ориентированного графа, не имеющего циклов (ациклического графа).

В результате топологической сортировки для вершин графа определяется такой порядок, что если их расположить на рисунке в соответствии с этим порядком сверху вниз, то дуги будут направлены только от верхних вершин к нижним.

Наиболее известны два способа топологической сортировки графа: алгоритмы Демукрона и алгоритм, применяющий поиск в глубину.

**Топологическая сортировка графа с помощью алгоритма поиска в глубину.**

Запустить алгоритм DFS, при выходе из вершины добавляя вершину в конец списка с ответом. После окончания алгоритма список с ответом развернуть в противоположном порядке.

**Алгоритм Демукрона вычисления порядковой функции сети**  
Этот метод состоит в вычислении порядковой функции сети и известен как алгоритм Демукрона. Предполагается, что вершины сети пронумерованы от 1 до ***n***.

1. **Потоки в сетях. Задача о максимальном потоке. Теорема Форда-Фалкерсона.**

***Сеть*** – это ориентированный граф , каждому ребру  которого поставлено в соответствие число , называемое пропускной способностью, а также выделено две вершины - исток и  - сток, .

***Поток*** – это функция ,  обладающая тремя свойствами:

1. ;

2.  (кососимметричность);

3. , 

***Величина потока*** это

***Разрез  сети***  называется разбиение множества  на две части  и  такое, что , , , .

***Пропускная способность разреза*** ****** – это сумма пропускных способностей дуг соединяющих вершины в  и .

***Минимальный разрез сети*** – это разрез с минимальной пропускной способностью.

Теорема Форда-Фалкерсона: В любой сети максимальная величина потока из истока  в сток  равна минимальной пропускной способности разреза отделяющего  от .

1. **Задачи нелинейного программирования. Основные алгоритмы решения.**

Нелинейное программирование – раздел математического программирования, изучающий методы решения экстремальных задач с нелинейной целевой функцией и (или) областью допустимых решений, определенной нелинейными ограничениями. К нелинейному программированию относят квадратичное, дробное, выпуклое, дискретное, целочисленное и геометрическое программирование.

В общем виде задачу нелинейного программирования можно сформулировать так:

*F*(*х*)→min (max)

Среди большого числа вычислительных алгоритмов нелинейного программирования значительное место занимают:

* различные варианты градиентных методов (метод проекции градиента, метод условного градиента и т. п.);
* методы штрафных функций;
* методы барьерных функций;
* метод модифицированных функций Лагранжа и др.

1. **Постановка задачи векторной оптимизации.**

Эффективность функционирования экономической системы оценивается, как правило, несколькими критериями. Математической формой критерия эффективности в оптимизационных экономико-математических задачах является целевая функция.

Пусть имеется  критериев, которые можно записать в виде целевых функций , где . Поскольку , то для простоты в дальнейшем будем предполагать, что все целевые функции максимизируются. Задача многокритериальной оптимизации в этом случае запишется

;.

Если точки максимума , определенные при решении задач по каждому критерию  не совпадают, то решение задачи (1)-(3) может быть только компромиссным. В области допустимых значений задачи находится область компромиссов. При перемещении из одной точки области компромиссов в другую, невозможно одновременное улучшение всех критериев. Принадлежащие области компромиссов планы называются эффективными или оптимальными по Парето (по имени итальянского экономиста, впервые сформулировавшего проблему многокритериальной оптимизации и принцип оптимальности).

План  оптимален по Парето, если он допустим и не существует другого плана  для которого



и хотя бы для одного критерия выполняется строгое неравенство.

К задачам векторной оптимизации приходят в следующих случаях:

1. Качество моделируемого процесса нужно оценить с точки зрения нескольких показателей. Это могут быть прибыль, себестоимость, рентабельность и т.д.
2. Моделируемые процесс представляет собой составляющую нескольких процессов (частей), и каждая из этих частей имеет свой критерий качества.
3. Моделируемые процесс расчленяется на несколько шагов и на каждом шаге его качество определяется своей функцией. (Например, на отдельных временных промежутках)
4. **Методы решения задач векторной оптимизации.**

Методы решения задач многокритериальной оптимизации можно подразделить на четыре группы:

* + - * методы, основанные на свертывании критериев;
      * методы, использующие ограничения на критерии;
      * методы целевого программирования;
      * методы, основанные на отыскании компромиссного решения.

Вместо исходной многокритериальной задачи в соответствии с выбранным методом, формируется замещающая задача. В состав замещающей задачи входит один критерий, а к исходной системе ограничений добавляется одно или несколько дополнительных ограничений. Решение замещающей задачи называется субоптимальным.