МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

 **«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Кафедра математических и компьютерных методов**

**Курсовая работа**

ОПТИМИЗАЦИЯ МОДЕЛЕЙ СЕТЕВОГО ПЛАНИРОВАНИЯ И УПРАВЛЕНИЯ.

Работу выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Загородный Г.Ю.

 (подпись, дата)

Факультет экономический

Направление 27.03.03 Системный анализ и управление

Научный руководитель

доцент кафедры МКМ, канд. эконом. наук, доцент \_\_\_\_\_\_\_\_\_\_\_\_\_\_Г.Н.Библя

 (подпись, дата)

Нормоконтролер

преподаватель
кафедры МКМ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Руденко С.Ю.

 (подпись, дата)

Краснодар 2017

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

 **«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Кафедра математических и компьютерных методов**

**ЗАДАНИЕ**

на курсовую работу

Студенту \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Загородному Г.Ю.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ группы 113 направления подготовки 27.03.03 Системный анализ

**Тема курсовой работы: «Оптимизация моделей сетевого планирования и управления»**

 **Цель**: Изучить сущность моделей сетевого планирования, а так же управления и их оптимизации. Выявить критерии оптимизации.

 **Основные вопросы, подлежащие разработке (исследованию)**:

 1) Теоретический обзор современных подходов, методов и алгоритмов оптимизации моделей сетевого планирования и управления;

 2) Анализ различных алгоритмов решений задач оптимизации моделей СПУ;

 3) Решение задач оптимизации моделей СПУ.

 **Основная литература**:

1. Зыков А.А. Основы теории графов. – М.: Наука, 2007. – 384 c.
2. Майника Э. Алгоритмы оптимизации на сетях и графах. Пер. с англ. - М.: Мир,2001. – 277 c.
3. Оре О. Теория графов. — 2-е изд.— М.: Наука, Главная редакция физико-математической литературы, 2010. – 336 c.
4. Уилсон Р. Введение в теорию графов. Пер. с анг. 2007. – 208 c.
5. Харари Ф. Теория графов / Пер.с англ. и предисл. В. П. Козырева. Под ред. Г. П. Гаврилова. Изд. 2-е. - М.: Едиториал УРСС, 2003. – 296 c.

Срок представления законченной работы 25 мая 2017 г.

Дата выдачи задания 01 февраля 2017 г.

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Г.Н. Библя /

Задание получил 01 февраля 2017 г.

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/

РЕФЕРАТ

Курсовая работа 31с., 6 рисунков, 2 таблица, 16 источников.

СИСТЕМНЫЙ АНАЛИЗ, ОПТИМИЗАЦИЯ, ТЕОРИЯ ГРАФОВ, ОСТОВНОЕ ДЕРЕВО, МАКСИМАЛЬНЫЙ ПОТОК, МИНИМАЛЬНЫЙ РЕЗЕРВ, СТРУКТУРНАЯ ОПТИМИЗАЦИЯ, EXCEL.

Объект исследования – сетевая модель графа.

Предмет исследования – методы оптимизации сетевых моделей.

Целью курсовой работы является изучить сущность моделей сетевого планирования, а так же управления и их оптимизации.

Метод исследования – методы структурной оптимизации, алгоритмы решения задач.

Основные результаты – получены оптимальные структуры остовного дерева, максимального потока сети.

Для выполнения поставленных задач была использована информация из учебных пособий и интернета.

Актуальность и практическая значимость предлагаемой работы заключается в том, что всегда возникает необходимость поисков эффективных способов планирования сложных процессов и проектов, которые в итоге позволяют создавать методы сетевого планирования и управления, а далее – принимать оптимальные управленческие решения.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc483955734)

[1. Основы теории графов 8](#_Toc483955735)

[1.1. Сущность и содержание теории графов 8](#_Toc483955736)

[1.2. Остовное дерево. Задача построения остовного дерева минимального веса 11](#_Toc483955737)

[2. Моделирование системы с помощью графов 14](#_Toc483955738)

[2.1. Формулировка алгоритма и решение задачи нахождения минимального остовного дерева 14](#_Toc483955739)

[2.2. Формулировка и решение задачи поиска кратчайшего пути 15](#_Toc483955740)

[2.3. Формулировка и решение задачи поиска максимального потока и минимального резерва сети 19](#_Toc483955741)

[3. Использование среды MS Excel для решения задачи: минимального остовного дерева, задачи поиска максимального потока 23](#_Toc483955742)

[3.1. Нахождение минимального остовного дерева 23](#_Toc483955743)

[3.2. Нахождение максимального потока в Excel. 26](#_Toc483955744)

[ЗАКЛЮЧЕНИЕ 29](#_Toc483955745)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 30](#_Toc483955746)

# ВВЕДЕНИЕ

Решение хозяйственных задач непрерывно связано с выполнением ряда работ, действий, операций и мероприятий. Некоторые из них можно выполнять параллельно и одновременно, а другие – исключительно в определенной последовательности. К примеру, чтоб начать какое-либо производство нового изделия, необходимо, прежде всего, выработать его конструкцию и технологию производства, а так же выполнять четыре вида одновременных работ:

− планировать размещение оборудования, а так же рассчитывать требуемые площади и строить помещения;

− проектировать, заказывать, получать и монтировать необходимое оборудование;

− заключать договора с различными предприятиями о поставках нужных материалов, сырья, а так же комплектующих деталей;

− набирать и подготавливать кадры будущих работников.

В условиях нашей современности необходимо разрабатывать и использовать относительно простые и эффективные методы руководства комплексными разработками.

Актуальность работы заключается в том, что всегда возникает необходимость поисков эффективных способов планирования сложных процессов и проектов, которые в итоге приводят к созданию методов сетевого планирования и управления.

Эффект, достигаемый за счет применения СПУ, обусловлен в первую очередь внесением строгих логических элементов в формирование плана, позволивших привлечь для анализа и синтеза планов реализации проектов современный математический аппарат и средства вычислительной техники.

В силу универсальности СПУ этот аппарат используется для формирования планов строительной индустрии во всех видах строительства, в индивидуальном и мелкосерийном производстве, в научно-исследовательских, опытно-конструкторских и проектных организациях, в производстве кинофильмов, в горнодобывающей промышленности и геологоразведочных работах.

Объектом управления в системах СПУ является коллектив, располагающий определенными ресурсами и выполняющий комплекс работ, призванный обеспечить достижение намеченной цели. Метод СПУ позволяет в любых, даже самых сложных ситуациях, быстро принимать наиболее правильные решения, выявить резервы времени и средств на одних участках работы и перебросить их на другие, более напряженные.

Важной особенностью систем СПУ является системный подход к вопросам организации управления, согласно которому коллективы исполнителей, принимающих участие в проекте и объединенные общностью поставленной перед ними задачи, рассматриваются как звенья единой сложной организационной системы.

Для отображения процесса выполнения проекта и управления им в системах СПУ используется сетевая модель.

Можно определить следующие цели работы:

1. изучить сущность моделей сетевого планирования, а так же управления;
2. выявить критерии оптимизации этих моделей;
3. из всего изложенного сделать вывод.

Объектом исследования выступают сетевые модели планирования и управления.

Предмет исследования – методы оптимизации сетевых моделей.

Теоретической и методологической основой написания работы послужили научные труды как российских, так и зарубежных деятелей разных сфер науки.

Структура курсовой работы обусловлена задачами, целью и характером объекта и предмета исследования.

Работа включает в себя введение, теоретическую главу и практическую часть, заключение и список использованных источников.

Общий объем работы составил − 31 страниц.

# 1. Основы теории графов

## 1.1. Сущность и содержание теории графов

Теория графов − это область дискретной математики, особенностью которой является геометрический подход к изучению объектов. Теория графов находится сейчас в самом расцвете. Часто её относят к топологии (из-за того, что во многих случаях рассматривают только лишь топологические свойства графов), но она пересекается с разными разделами теории множеств, комбинаторной математики, алгебры, геометрии, теории матриц, теории игр, математической логики и многих других математических дисциплин. Ключевой объект теории графов – граф, а так же его обобщения[1].

1. Граф-пара объектов G = (x, Г),

Где x-конечное множество, а Г – конечное подмножество прямого произведения х \* Х. Причем х называют множеством вершин, и Γ-множество дуг графа G.

2. Любое конечное множество вершин (точек), некоторые из которых попарно соединены стрелками, (в теории графов эти стрелки называются ду-гами), можно рассматривать как граф.

3. Граф называется ориентированным, если в наборе Γ все пары упорядочены[2].

4. Дуга-ребро ориентированного графа.

5. Если у графа нет ребер, то граф называется вырожденным

6. Вершина x называется инцидентной краю G(ребру), если ребро соединяет эту вершину с какой-то другой вершины.

7. Подграфом G (V1, Е1) графа G (V, Е) является граф с набором вершин V1, V и множеством дуг (ребер), Е1, Е, таких, что каждое ребро (дуга) из Е1 инцидентно (инцидента) только вершине V1. Другими словами, подграф включает в себя некоторые вершины исходного графа и некоторые pебра (только тех, чьи оба конца входят в подграф).

8. Подграф, порожденный множеством вершин U – подграф, у которого у которого множество вершин – U, который содержит только те ребра, концы которых, причем обоих, входят в U.

9.Остовной подграф – подграф, у которого множество вершин(его) совпадает с множеством вершин самого графа.

10. Если существует ребро, соединяющее вершины, то они называются смежными.

11. Если существует вершина одновременно инцидентная G1 и G2, то грани (ребра) G1 и G2 являются смежными.

12. Каждый график может быть представлен множеством точек в пространстве, соответствующих вершинам, соединённых линиями, соответствующими ребрам (в дугах стрелками обычно указывают направление). – данное представление является диаграммой укладки.

13. Доказано, что в 3-мерном пространстве любой граф можно представить в виде укладки таким образом, что линии, соответствующие дугам (ребрам) во внутренних точках не пересекаются. Для 2-мерного пространства данное утверждение является ложным. Графы, которые допускают представление в виде укладки в 2-мерном пространстве называют планарными (плоскими). Иными словами, планарный граф- граф, который можно изобразить на плоскости так, что его ребра не пересекаются.

14. Грань графа, представленного на определенной поверхность –ограниченная ребрами графа часть поверхности. Это понятие грани, в сущности, совпадает с понятием грани многогранника. Поверхность многогранника выступает в качестве поверхности в этом случае. Если многогранник является выпуклым, он может быть нарисован на плоскости, все грани, при этом, будут сохранены. Например, это можно изобразить так: растягиваем одну из граней многогранника, "расплющиваем" многогранник, так что он вписывается в эти грани. Плоский граф – то, что мы получаем в результате. Грань, что мы протянули "исчезнет", но ей будет соответствовать грань, состоящая из части плоскости, ограничивающей граф. Так можно говорить о ребрах, вершинах и гранях многогранника, а оперировать соответствующими понятиями для плоского графа[4].

15. Пустой граф – граф без ребер. Граф называется полным, в котором смежные каждые две вершины.

16. Маршрут длины N – конечная последовательность необязательно различных ребер R1, R2,..., RN, , если существует последовательность В1, В2, ... вн, не обязательно различных вершин, таких, что Ri = (Вi-1, Вi).

17. Маршрут замкнут, если они совпадают.

18. Цепь – маршрут, где все грани( ребра) различны попарно.

19. Цикл – замкнутый маршрут, ребра которого, причем все, различны. Простой цикл (цепь) – различны все вершины цикла(цепи).

20. Простая цепь – маршрут, где попарно различны все вершины.

21. Простой цикл – все вершины различны попарно, кроме последней и первой.

22. Связный граф – существует путь для любых двух вершин, соединяющий данные две вершины[5].

23. Любой максимальный связанный подграф графа G является компонентой связности. Граф несвязный имеет хотя бы две компоненты связности.

24. К-связный граф – граф в котором удаление не менее K ребер (вершин) приводит к потере связности.

25. Обход графа – маршрут, включающий в себя все ребра и вершины и обладающий определенными свойствами.

26. Длина маршрута – количество ребер в порядке их прохождения. Длина кратчайшей простой цепи, соединяющей вершины Ri и Rj в графе G называется расстояние D (Ri, то Rj) между Ri и Rj.

27. Степень вершины - количество инцидентных ребер для вершины V обозначается F (C). По методам различных операций можно строить графы из более простых, разбивать графы на более простые, переходить от графа к более простому, и так далее. Среди одношаговых операций, наиболее распространенными являются добавление и удаление вершины или ребра, стягивание ребра, подразбиение ребра (т. е. замена ребра (y, c) на пару (y, q), (q , c), где q - новая вершина) и др. Известны двумшаговые операции: соединение, различные виды умножений графов, сложение и др. Эти операции используются для синтеза и анализа графов с заданными свойствами.

28. Изоморфные графы – два графа G1=(С1;E1), G2=(C2;W2), при условии существования взаимно-однозначного соответствия между множествами вершин C1 и C2 и между множествами рёбер W1 и W2, такое, при котором сохраниться отношение инцидентности. Изоморфизм, как понятие, для графов имеет наглядное толкование. Вообразим, что рёбра графов – эластичные нити, связывающие узлы – вершины. В таком случае, изоморфизм можно трактовать, как растяжение нитей и перемещение узлов.

29. Дерево – связный граф без циклов. При изображении различных иерархий, на практике, особенно часто возникают. Популярные генеалогические деревья – явный тому пример.

## 1.2. Остовное дерево. Задача построения остовного дерева минимального веса

Остовное дерево  — ациклический [связный](https://ru.wikipedia.org/wiki/%D0%A1%D0%B2%D1%8F%D0%B7%D0%BD%D1%8B%D0%B9_%D0%B3%D1%80%D0%B0%D1%84) [подграф](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B4%D0%B3%D1%80%D0%B0%D1%84) данного связного [неориентированного графа](https://ru.wikipedia.org/wiki/%D0%9D%D0%B5%D0%BE%D1%80%D0%B8%D0%B5%D0%BD%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D1%8B%D0%B9_%D0%B3%D1%80%D0%B0%D1%84), в который входят все его вершины. Неформально говоря, остовное дерево состоит из некоторого подмножества рёбер графа, таких, что из любой вершины графа можно попасть в любую другую вершину, двигаясь по этим рёбрам, и в нём нет циклов, то есть из любой вершины нельзя попасть в саму себя, не пройдя какое-то ребро дважды[6].



Рисунок 1 – Пример минимального остовного дерева в графе. Числа на ребрах обозначают вес ребер.

 Минимальное остовное дерево (или минимальное покрывающее дерево) в связанном взвешенном неориентированном графе — это остовное дерево этого графа, имеющее минимальный возможный вес, где под весом дерева понимается сумма весов входящих в него рёбер.

Задача о нахождении минимального остовного дерева часто встречается в подобной интерпретации: допустим, есть n городов, которые необходимо соединить дорогами, так, чтобы можно было добраться из любого города в любой другой (напрямую или через другие города). Разрешается строить дороги между заданными парами городов и известна стоимость строительства каждой такой дороги. Требуется решить, какие именно дороги нужно строить, чтобы минимизировать общую стоимость строительства.

Задача может быть поставлена в терминологии теории графов, как задача о нахождении минимального остовного дерева в графе, вершины которого представляют города, рёбра – это пары городов, между которыми можно проложить прямую дорогу, а вес ребра равен стоимости строительства соответствующей дороги.

Существует несколько алгоритмов для нахождения минимального остовного дерева (более подробно рассмотрим в пункте 2.1). Некоторые наиболее известные из них перечислены ниже[7]:

* Алгоритм Прима
* Алгоритм Краскала (или алгоритм Крускала)
* Алгоритм Борукви

Задача о назначениях. Алгоритм Мака.

В процессе управления производством зачастую возникают задачи назначения исполнителей на различные виды работ. Например, подбор кадров и назначение кандидатов на вакантные должности, распределение источников капиталовложений между различными проектами научно-технического развития, распределение самолетов между авиалиниями. Задача состоит в выборе элементов – по одному в каждом столбце, таких, что их сумма минимальна. Каждый человек должен быть полностью задействован, а каждое задание выполнено на 100%.

Алгоритм Мака основан на идее выбора в каждой строке минимального элемента. Тут применяется идея сложения (или вычитания) одного и того же значения со всеми элементами строки или столбца, чтобы распределить минимальные элементы строк по столбцам. Если в каждом столбце имеется ровно по одному подчеркнутому элементу, то подчеркнутые элементы – базис – определяют оптимальный выбор

# 2. Моделирование системы с помощью графов

## 2.1. Формулировка алгоритма и решение задачи нахождения минимального остовного дерева

На основе графических представлений возникли методы, которые позволяют ставить и решать вопросы оптимизации процессов организации, управления, проектирования и являются математическими методами в традиционном смысле.

Как говорилось в пункте 1.1, существует несколько алгоритмов для нахождения минимального остовного дерева. Рассмотрим наиболее известные более подробно:

1) Алгоритм Прима. На вход алгоритма подаётся связный неориентированный граф. Для каждого ребра задаётся его стоимость. В первую очередь берётся произвольная вершина и находится ребро, инцидентное данной вершине и обладающее наименьшей стоимостью. Найденное ребро и соединяемые им две вершины образуют дерево. Затем, рассматриваются рёбра графа, один конец которых — уже принадлежащая дереву вершина, а другой – нет; из этих рёбер выбирается ребро наименьшей стоимости. Выбираемое на каждом шаге ребро присоединяется к дереву. Таким образом, при выполнении каждого шага алгоритма, высота формируемого дерева увеличивается на 1. Рост дерева происходит до тех пор, пока не будут исчерпаны все вершины исходного графа[8].

2) Алгоритм Краскала (или алгоритм Крускала). Для начала текущее множество рёбер устанавливается пустым. После, пока это возможно, проводится следующая операция: из всех рёбер, добавление которых к уже имеющемуся множеству не вызовет появление в нём цикла, выбирается ребро минимального веса и добавляется к уже имеющемуся множеству. Когда таких рёбер больше нет, алгоритм завершён. Подграф данного графа, содержащий все его вершины и найденное множество рёбер, является его остовным деревом минимального веса[9].

3) Алгоритм Борувк. Выполнение алгоритма состоит из нескольких итераций, каждая из которых состоит в последовательном добавлении рёбер к остовному лесу графа, до тех пор, пока лес не превратится в [дерево](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%80%D0%B5%D0%B2%D0%BE_%28%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2%29), то есть, лес, состоящий из одной компоненты связности[10].

В псевдокоде, алгоритм можно описать так:

1. Изначально, пусть T — пустое множество рёбер (представляющее собой остовный лес, в который каждая вершина входит в качестве отдельного дерева).
2. Пока {\displaystyle T}T не является деревом (что эквивалентно условию: пока число рёбер в {\displaystyle T}T меньше, чем V – 1 , где {\displaystyle V}V — число вершин в графе): для каждой компоненты связности (то есть, дерева в остовном лесе) в подграфе с рёбрами {\displaystyle T}T, найдём самое дешёвое ребро, связывающее эту компоненту с некоторой другой компонентой связности. (Предполагается, что веса рёбер различны, или как-то дополнительно упорядочены так, чтобы всегда можно было найти единственное ребро с минимальным весом); добавим все найденные рёбра во множество {\displaystyle T}T.
3. Полученное множество рёбер {\displaystyle T}T является минимальным остовным деревом входного графа.

##

## 2.2. Формулировка и решение задачи поиска кратчайшего пути

Задача поиска кратчайшего пути – задача, где необходимо найти кратчайший путь между двумя вершинами (точками) на графе, при уменьшая сумму весов ребер, которые составляют этот путь.

Задача может быть определена и решена для ориентированного, смешанного и неориентированного графов. Самую легкую задачу, в данном случаи, представляет неориентированный граф, без учета направлений ребер, направление следует учитывать в ориентированных и смешанном графах[11].

В конкретной задачи, вес ребра также может представлять (в данном случаи – заменяться) на стоимость, скорость, расходы и т.п..

Задача поиска кратчайшего пути на графе может быть определена для неориентированного, ориентированного или смешанного графа. Рассмотрим постановку задачи в наиболее простом виде – для неориентированного графа. Для смешанного и ориентированного графа дополнительно должны учитываться направления ребер[12].

Граф – совокупность непустого множества ребер и вершин (а также – наборов пар вершин). Если две вершины на графе соединяются общим ребром – они смежны. Путь в неориентированном графе представляет собой последовательность вершин O=(C1,C2,…Cn)∈ C×C×…C {\displaystyle P=(v\_{1},v\_{2},\ldots ,v\_{n})\in V\times V\times \ldots \times V}, таких что {\displaystyle v\_{i}}Ci смежна с Ci+1{\displaystyle v\_{i+1}} для 1≤ i< n.{\displaystyle 1\leq i<n}111 Данный путь {\displaystyle P}O является путём длиной {\displaystyle n} n из вершины C1 {\displaystyle v\_{1}}vв Cn {\displaystyle v\_{n}}({\displaystyle i}ii показывает номер вершины пути, к нумерации вершин на графе не имеет никакого отношения)[13].

Пусть wij{\displaystyle e\_{i,j}} — ребро соединяющее две вершины: Ci {\displaystyle v\_{i}}и Cj{\displaystyle v\_{j}}. Имеется весовая функция d: W → E{\displaystyle f:E\rightarrow \mathbb {R} }, отображающая ребра на их веса, их значение выражаются действительными числами, и неориентированный граф G{\displaystyle G}. В таком случаи, кратчайшим путём из вершины C {\displaystyle v}в вершину Cd {\displaystyle v'}будет называться путь O=(C1,C2,…Cn) {\displaystyle P=(v\_{1},v\_{2},\ldots ,v\_{n})} (где{\displaystyle v\_{1}=v} C=C1 и Cn = Cd {\displaystyle v\_{n}=v'}), имеющий минимальное значение суммы  {\displaystyle \sum \_{i=1}^{n-1}f(e\_{i,i+1}).}
$$\sum\_{i=1}^{n-1}d\left(wi,j+1\right)$$

 При условии, что все ребра в графе имеют единичный вес, задача сводится к нахождению наименьшего количества огибаемых ребер.

Встречаются различные постановки задачи о кратчайшем пути:

- задача о кратчайшем пути в заданный пункт назначения. Требуется найти кратчайший путь в указанную вершину назначения r, начинающийся в каждой из вершин графа (кроме r). Поменяв направление всех ребер, принадлежащих графу, ее можно свести к задаче о единой исходной вершине (в которой из заданной вершины осуществляется поиск кратчайшего пути во все остальные).

- задача о кратчайшем пути между заданной парой вершин. Необходимо найти кратчайший путь из заданной вершины Y в заданную вершину C.

- задача о кратчайшем пути между всеми парами вершин. Необходимо найти кратчайший путь из каждой вершины Y в каждую вершину C. Данную задачу можно решить, воспользовавшись алгоритмом, предназначенным для решения задач об одной исходной вершине, но зачастую она решается быстрее.

В разных формулировках задач, роль длины ребра могут играть не только сами длины, но и стоимость, время, расходы, объем затрачиваемых ресурсов (финансовых, материальных, топливно-энергетических и т. п.) либо же иные характеристики, касающиеся прохождения каждого ребра. Так, задача находит практическое применение в огромном количестве областей (начиная от информатика и экономик, и заканчивая географией).

Из-за существования множества различных вариаций постановок данной задачи, существуют наиболее популярные алгоритмы для решения задачи поиска кратчайшего пути на графе:

- алгоритм Дейкстры, находящий кратчайший путь до всех вершин графа из одной. Работает лишь для графов, не имеющих ребер отрицательного веса.

- алгоритм Беллмана — Форда, находящий кратчайшие пути до всех вершин от одной во взвешенном графе. Вес ребер может быть отрицательным.

- алгоритм поиска A\*, находящий с наименьшей стоимостью маршрут от начальной вершины к целевой(конечной), используя алгоритм поиска по первому наилучшему совпадению на графе.

- алгоритм Флойда – Уоршелла, находящий кратчайшие пути между всеми вершинами взвешенного ориентированного графа.

- алгоритм Джонсона, находящий кратчайшие пути между всеми парами вершин взвешенного ориентированного графа.

- алгоритм Ли (он же - волновой алгоритм), находящий путь между вершинами a и r графа (a не совпадает с r), содержащий минимальное количество промежуточных ребер(вершин). Основным его применением, без сомнения, являются трассировки электрических соединений на кристаллах микросхем а также – на печатных платах. Еще находит свое применение в поиске кратчайшего расстояния на карте в стратегических играх.

- поиск кратчайшего пути на основе алгоритма Килдала.

Рассмотрим один из них - алгоритм Дейкстры[14]

Как было описано выше, он находит кратчайшее расстояние от одной из вершин графа до всех остальных, работает только для графов без рёбер отрицательного веса.

Пример: пусть дан взвешенный ориентированный граф G (C, W) без петель и дуг отрицательного веса, нужно найти кратчайшие пути от некоторой вершины s графа G до всех остальных вершин этого графа.

Алгоритм:

1) Всем вершинам из C сопоставим метки – минимальное известное расстояние от этих вершин до s. Работа алгоритма пошагова – он «посещает» одну вершину на каждом шаге и пытается уменьшать метки. Алгоритм завершает свою работу, как только посещены все вершины.

2) Инициализация. Метка самой вершины s полагается равной 0, метки остальных вершин – бесконечности. Это показывает то, что расстояния от s до остальных вершин в данный момент неизвестны. Все вершины графа отмечаются как не посещённые.

3) Шаг алгоритма. Когда все вершины были посещены, алгоритм завершается. Иначе, из ещё не посещённых вершин отбирается вершина y, имеющая наименьшую метку. Мы рассматриваем все возможные маршруты, в которых y – предпоследний пункт. Соседями этой вершины назовем вершины, в которые ведут рёбра из y. Для каждого отдельного соседа вершины y, кроме отмеченных как посещённые, рассмотрим новую длину пути, равную сумме значений текущей метки y и длины ребра, соединяющего y с этим соседом. В случаи, если полученное значение длины меньше значения метки соседа, подставим полученное значение длины. Рассмотрев всех соседей, пометим вершину y, как посещенную и вновь повторим шаг алгоритма.

## 2.3. Формулировка и решение задачи поиска максимального потока и минимального резерва сети

Задача о максимальном потоке заключается в нахождении такого потока по транспортной сети, что сумма всех потоков из истока максимальна (сумма потоков в сток)[15].

Частным случаем более сложных задач является задача о максимальном потоке.

Определение: дана транспортная сеть B=(C,W) с источником s∈C, стоком r∈C и пропускными способностями c. Величина потока – сумма потоков из источника |f|=∑c∈cfac и она равна сумме потоков в сток $\sum\_{w\in c=0}^{}f(q,r)$.[16]

Следующая таблица перечисляет некоторые алгоритмы решения задачи.

Таблица №1 Алгоритмы решения

|  |  |  |
| --- | --- | --- |
| Метод | Сложность | Описание |
| Линейное программирование | Влияет выбор конкретного алгоритма.Для симплекс метода экспоненциальна. | Сформулировать задачу о максимальном потоке, как задачу линейного программирования. Потоки по рёбрам – переменные, сохранение потока, ограничение пропускной способности – ограничения. |
| Алгоритм Форда-Фалкерсона | Зависит от алгоритма поиска увеличивающего пути.Требует O(max| f |) таких поисков. | Найти любой увеличивающий путь. Увеличить поток по всем его рёбрам на минимальную из их остаточных пропускных способностей. Пока увеличивающий путь есть – повторять. Работает лишь для целых пропускных способностей. Иначе, будет работать бесконечно долго, не приходя к правильному ответу. |
| Алгоритм Эдмонса-Карпа | O(VE²) | Выполняем алгоритм Форда-Фалкерсона, каждый раз выбирая кратчайший увеличивающий путь (находится поиском в ширину). |
| АлгоритмДиницы | O(V2E)O(V√E) для единичных пропускных способностей(O(VElog(V)) с использованием деревьев Слетора и Тарьяна. | Усовершенствование алгоритма Эдмондса-Карпа (найден раньше). Применяя поиск в ширину, на каждой итерации, определяем расстояния от всех вершин в остаточной сети до источника. Строим граф, содержащий только такие рёбра остаточной сети, на которых это расстояние растёт на 1. Исключаем из графа все тупиковые вершины с инцидентными им рёбрами, пока все вершины не станут не тупиковыми. (Тупиковая вершина – в которую не входит ни одно ребро или из которой не исходит ни одного ребра, кроме источника и стока) На полученном графе находим кратчайший увеличивающий путь (им будет любой путь из a в r). Убираем из остаточной сети ребро с минимальной пропускной способностью, вновь убираем тупиковые вершины, и так до тех пор, пока увеличивающий путь есть. |
| Алгоритм проталкивания предпотока | O(V2E) | Вместо потока оперирует с предпотоком. Отличие в том, что для любой вершины u, кроме источника и стока, совокупность входящих в неё потоков для потока должна быть строго 0 (условие сохранения потока), а для предпотока — неотрицательной. Эта сумма – избыточный поток в вершину, а вершина с положительным избыточным потоком – переполненная. Для каждой вершины алгоритм сохраняет дополнительную характеристику, высоту, которая является целым неотрицательным числом. Алгоритм пользуется двумя операциями: проталкивание: поток по ребру, идущему из более высокой в более низкую вершину, увеличивается на максимально возможную величину; и подъём: переполненная вершина, проталкивание из которой невозможно из-за недостаточной высоты, поднимается. Проталкивание возможно, когда ребро принадлежит остаточной сети, ведёт из более высокой вершины в более низкую, и исходная вершина переполнена. Подъём возможен, когда поднимаемая вершина переполнена, но ни одна из вершин, в которые из неё ведут рёбра остаточной сети, не ниже её. Он совершается до высоты на 1 большей, чем минимальная из высот этих вершин. Изначально высота источника V, по всем рёбрам, исходящим из источника, течёт максимально возможный поток, а остальные высоты и потоки нулевые. Операци проталкивания и подъёма выполняются до тех пор, пока это возможно. |
| Алгоритм“поднять в начало” | O(V³)O(VE log(V²/E)) с использованием динамических деревьев | Вариант прошлого алгоритма, специальным образом определяющий порядок операций проталкивания и подъёма. |

С помощью любого из этих алгоритмов можно найти такой поток по транспортной сети, что сумма всех потоков из истока максимальна. Многие из них взаимосвязаны и переплетаются между собой.

# 3. Использование среды MS Excel для решения задачи: минимального остовного дерева, задачи поиска максимального потока

## 3.1. Нахождение минимального остовного дерева

Задача о нахождении минимального остовного дерева часто встречается в подобной постановке: есть n городов, через которые можно проложить маршрут так, чтобы можно было добраться из любого города в любой другой (напрямую или через другие города). Требуется найти такой маршрут, чтобы стоимость проезда была максимальной.

Эта задача может быть сформулирована в терминах теории графов как задача о нахождении минимального остовного дерева в графе, вершины которого представляют города, рёбра - это пары городов, между которыми есть маршрут, а вес ребра равен стоимости проезда по соответствующему маршруту.

Пусть имеется связный неориентированный граф G, на ребрах которого задана весовая функция c (e). Связный подграф графа G, являющийся деревом и содержащий все его вершины, называют покрывающим деревом (spanning-tree) этого графа (иногда используют термин остовное дерево или остов). Весом остовного дерева будем считать сумму весов всех его ребер. Тогда возникает задача о нахождении максимального покрывающего дерева, т.е. такого, что его вес наибольший, либо равен весу любого из покрывающих деревьев для данного графа.



Рисунок 2 – алгоритм Крускала вычисления MST.

Найдем минимальное остовное дерево с помощью алгоритма Крускала. Список ребер графа задан в произвольном порядке (рисунок 2 левый список ребер). Первый шаг алгоритма Крускала – сортировка ребер по их весу (правый список ребер на рисунке 2). После этого необходимо просмотреть ребра этого списка в порядке возрастания их весов, добавляя в MST ребра, которые не создают в нем циклов. Сперва мы добавляем ребро 5-3 (самое короткое ребро), потом 7-6 (слева), затем 0-2 (справа вверху) и 0-7 (справа, вторая диаграмма сверху). Ребро 0-1 со следующим весом (по величине) создает цикл, поэтому не добавляется в дерево. Ребра, которые не включаются в MST, выделены в отсортированном списке серым цветом. Затем мы добавляем ребро 4-3 (справа, третья диаграмма сверху). Далее мы отбрасываем ребро 5-4, поскольку оно образует цикл, и потом добавляем 7-4 (справа внизу). Когда MST-дерево готово, любое ребро с большим весом образует цикл и в связи с этим будет отброшено (когда в MST будут включены V— 1 ребер, алгоритм останавливается). Эти ребра в отсортированном списке помечены звездочками.

Ребра добавляются в MST-дерево в порядке возрастания их длины — таким образом, лес содержит вершины, соединенные друг с другом относительно короткими ребрами. В любой момент выполнения алгоритма каждая вершина расположена ближе к некоторой вершине своего поддерева, чем к любой другой вершине, не входящей в это дерево.

## 3.2. Нахождение максимального потока в Excel.

 Запишем формулы:

 (1)

$\sum\_{\left(i, j\in M \right)}^{}fij$ - $\sum\_{l, i\in M)}^{}flj$ = 0, *i ≠ N, i ≠ s, t;*  (2)

$\sum\_{t, j\in M)}^{}ftj$ *-* $\sum\_{(l, t\in M)}^{}flt$ *= - f*max , *i=t;* (3)

*fij ≤ cij , (i, j)* $\in $ *M;* (4)

*fij ≥ 0, (i, j)* $\in $ *M.* (5)

Дана сеть:


Рисунок 3.

Таблица №2 дано.



Определить:

1. Максимальный поток s – t;
2. Минимальный резерв сети;



Рисунок 4 – ввод исходных данных прямой задачи нахождения максимального потока.

В ячейки столбца А введены номера начальных узлов дуг, в столбец В – конечных узлов. В ячейки столбца С введены пропускные способности дуг. Ячейки столбца F зарезервированы для определения потоков по дугам сети. В ячейку F8 помещена как целевая функция (формула 1), представляющая собой сумму потоков, выходящих из источника по всем дугам. В столбец J введены левые части УСП(формулы 2 и 3). В ячейки столбца К введены фиксированные потоки, равные 0 для всех узлов кроме источника и стока. В стоке фиксированный поток определен противоположным по знаку значению целевой функции.

В окне “Поиск решения” значение целевой ячейки определяем максимальным, ячейки F1\F7 определяем изменяемыми. В поле “Ограничения ” первое неравенство соответствует формуле (4), второе неравенство – формуле (5), третье неравенство – формулам (2), (3).



Рисунок 5 – окно “Поиск решения” для прямой задачи нахождения максимального потока.

После запуска на выполнение получаем решение. Максимальный поток в сети равен 6, что показывает значение целевой ячейки F8. В ячейках F1\F6 определены потоки по дугам, дающие такое значение максимального потока.



Рисунок 6 – Решение задачи нахождения максимального потока.

# ЗАКЛЮЧЕНИЕ

В данной курсовой работе были изучены и исследованы модели сетевого планирования и управления на примере экономических задач.

В ходе исследования оптимизационных методов я выявил основные критерии оптимизации этих моделей, научился решать задачи, связанные с оптимизацией деятельности предприятия, рационального распределения ресурсов. Узнал какие экономические задачи и как решаются в среде MS Excel, вник в их суть и привел примеры решений.

Данная курсовая работа наглядно показала эффективность использования алгоритмов для решения различных задач, связанных с оптимизацией экономической деятельности: нахождения минимального остовного дерева, поиска кратчайшего пути, поиска максимального и минимального резерва сети. Мной были исследованы и изучены следующие методы и алгоритмы: алгоритм Прима, алгоритм Краскала, алгоритм Борувк, алгоритм Дейкстры, алгоритм Беллмана – Форда, алгоритм Флойда – Уоршелла, алгоритм Джонсона, алгоритм Ли (волновой), множество алгоритмов поиска максимального потока и минимального резерва сети (таблица 1).

Оптимизация моделей сетевого планирования и управления оказывает огромное влияние как на производственные процессы, так и на экономику в целом, ведь инструменты, по средствам которых происходит оптимизация моделей сетевого планирования и управления, позволяют нам оптимизировать логистику, и, как следствие, сократить издержки предприятия, что без сомнения является важным вопросом.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. А. В. Арсирий, Б. Ф. Трофимов, Е.М. Страхов. Сетевые модели: Методические указания и варианты заданий для самостоятельной работы, 2013. – 42с.
2. Алексеев В.Е., Таланов В.А. Глава 3.4. Нахождения кратчайших путей в графе. Нижний Новгород: Издательство Нижегородского гос. университета, 2005. – 307 c.
3. Галкина В.А. Глава 4. Построение кратчайших путей в ориентированном графе. Дискретная математика. Комбинаторная оптимизация на графах, 2003. – 221 c.
4. Евстигнеев В. А. Глава 3. Итеративные алгоритмы глобального анализа графов. Пути и покрытия. Под ред. А. П. Ершова. — Москва: Наука. Главная редакция физико-математической литературы, 1995. – 119 c.
5. Зыков А.А. Основы теории графов. – М. : Наука, 2007. – 384 c.
6. Кормен Т. Х. и др. Часть VI. Алгоритмы для работы с графами // Алгоритмы: построение и анализ = Introduction to Algorithms. — 2-е изд. — М.: Вильямс, 2006. – 1296 c.
7. Кормен, Т., Лейзерсон, Ч., Ривест, Р., Штайн, К. Алгоритмы: построение и анализ = Introduction to Algorithms / Под ред. И. В. Красикова. — 2-е изд. — М.: Вильямс, 2005. – 1296 c.
8. Кормен, Т., Лейзерсон, Ч., Ривест, Р., Штайн, К. Глава 23. Минимальные остовные деревья // Алгоритмы: построение и анализ = Introduction to Algorithms / Под ред. И. В. Красикова. — 2-е изд. — М.: Вильямс, 2005.– 1296 c.
9. Кристофидес Н.Теория графов. Алгоритмический подход. М.: Мир, 2008. – 432 c.
10. Майника Э. Алгоритмы оптимизации на сетях и графах. Пер. с англ. - М.:Мир,2001. – 277 c.
11. Оре О. Теория графов.— 2-е изд.— М.: Наука, Главная редакция физико-математической литературы, 2010. – 336 c.
12. Свами М., Тхуласираман К. Графы, сети и алгоритмы. М: Мир, 2004. – 454 c.
13. Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн. Алгоритмы: построение и анализ = Introduction to Algorithms. — 2-е изд. — М.: «Вильямс», 2006. –1296 c.
14. Уилсон Р. Введение в теорию графов. Пер. с анг. 2007. – 208 c.
15. Харари Ф. Теория графов / Пер.с англ. и предисл. В. П. Козырева. Под ред. Г. П. Гаврилова. Изд. 2-е. - М.: Едиториал УРСС, 2003. – 296 c.
16. Э. Майника. Алгоритмы оптимизации на сетях и графах. М.: Мир, 1981. – 323 c.