Дискретная математика

Лекции №1,2,3

Раздел №1. Основы теории графов

Содержание: Введение. Возникновение дискретной математики. Задачи на графах. Понятие графа, методы представления графа. Примеры. Свойства матриц графа. Изоморфизм графов. Мультиграф, орграф, взвешенный граф. Степень вершины. Свойства степеней вершин графа. Полный граф. Дополнение к графу. Цепь и путь в графе. Примеры. Связность графа. Циклы графа. Деревья. Свойства деревьев. Остовное дерево графа. Построение остовного дерева графа. Примеры.

Введение в курс

Целью освоения дисциплины «Дискретная математика» является:

– формирование математической и информационной культуры студента;

– привитие понимания универсального характера дискретных структур данных, понимания роли и места дискретной математики в системе наук;

– развитие абстрактного мышления, общей математической и информационной культуры мышдения.

	Название разделов и тем	Лекции	Практики
1	**Теория графов**		
1.1	Основы теории графов	8	6
1.2	Эйлеровы и Гамильтоновы графы	4	4
2	**Прикладные задачи дискретной математики**		
2.1	Прикладные задачи теории графов	8	10
2.2	Основы комбинаторики	4	4
Итого по дисциплине		**24**	**24**

Дискретная математика относятся к числу основных разделов современной теоретической информатики. Знание основ дискретной математики является важной составляющей общей информационной культуры выпускника. Эти знания необходимы как при проведении теоретических исследований в различных областях математики и информатики, так и при решении практических задач из разнообразных прикладных областей, таких как программирование, экономика, теоретическая информатика, обработка и передача данных, криптография и др.

Используемая литература

1. Папшев, С. В. Дискретная математика. Курс лекций для студентов естественнонаучных направлений подготовки : учебное пособие / С. В. Папшев. — Санкт-Петербург : Лань, 2019. — 192 с. — ISBN 978-5-8114-3292-9. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/113904 (дата обращения: 20.01.2023). — Режим доступа: для авториз. пользователей.

V1 Элементы комбинаторики 116-151

V11 Элементы теории графов 152-183

2. Алексеев, В. Е. Графы и алгоритмы: структуры данных. Модели вычислений : [16+] / В. Е. Алексеев, В. А. Таланов. – 2-е изд., испр. – Москва : Национальный Открытый Университет «ИНТУИТ», 2016. – 154 с. : ил. – (Основы информационных технологий). – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=428827 (дата обращения: 20.01.2023). – Библиогр. в кн. – ISBN 5-9556-0066-3. – Текст : электронный.

3. Ерусалимский, Я.М. Дискретная математика. Теория и практикум : учебник / Я.М. Ерусалимский. — Санкт-Петербург : Лань, 2018. — 476 с. — ISBN 978-5-8114-2908-0. — Текст : электронный // Электронно-библиотечная система «Лань» : [сайт]. — URL: https://e.lanbook.com/book/106869.

4. Кожухов, С.Ф. Сборник задач по дискретной математике : учебное пособие / С.Ф. Кожухов, П.И. Совертков. — 2-е изд., стер. — Санкт-Петербург : Лань, 2018. — 324 с. — ISBN 978-5-8114-2588-4. — Текст : электронный // Электронно-библиотечная система «Лань» : [сайт]. — URL: https://e.lanbook.com/book/102606.

5. Бережной, В. В. Дискретная математика : учебное пособие : [16+] / В. В. Бережной, А. В. Шапошников ; Северо-Кавказский федеральный университет. – Ставрополь : Северо-Кавказский Федеральный университет (СКФУ), 2016. – 199 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=466802 (дата обращения: 20.01.2023). – Библиогр. в кн. – Текст : электронный.

6. Васильева, А. В. Дискретная математика : учебное пособие : [16+] / А. В. Васильева, И. В. Шевелева ; Сибирский федеральный университет. – Красноярск : Сибирский федеральный университет (СФУ), 2016. – 128 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=497748 (дата обращения: 20.01.2023). – Библиогр.: с. 125. – ISBN 978-5-7638-3511-3. – Текст : электронный.

7. Костюкова, Н. Графы и их применение : [16+] / Н. Костюкова. – Москва : Национальный Открытый Университет «ИНТУИТ», 2016. – 148 с. : ил. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=429066 (дата обращения: 21.01.2023). – ISBN 978-5-9556-0069-7. – Текст : электронный.

8. Шевелев, Ю.П. Дискретная математика : учебное пособие / Ю.П. Шевелев. — 4-е изд., стер. — Санкт-Петербург : Лань, 2019. — 592 с. — ISBN 978-5-8114-4284-3. — Текст : электронный // Электронно-библиотечная система «Лань» : [сайт]. — URL: https://e.lanbook.com/book/118616.

Часть четвертая. Комбинаторика. 402-471
Часть пятая. Теория графов. 472-543 (все о графах, кроме алгоритмики)

9. Окулов, С. М. Дискретная математика: теория и практика решения задач по информатике : учебное пособие : [12+] / С. М. Окулов. – 4-е изд. – Москва : Лаборатория знаний, 2020. – 425 с. : ил. – (Педагогическое образование). – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=222848 (дата обращения: 20.01.2023). – Библиогр.: с. 414-415. – ISBN 978-5-00101-684-7. – Текст : электронный.

Возникновение дискретной математики. Задачи на графах

В настоящее время теория графов стала очень популярной среди учителей, школьников и студентов. Это связано с тем, что при помощи этой теории можно довольно просто решать большой круг самых разнообразных математических задач. На языке графов условия задач приобретают завидную наглядность, что упрощает их анализ. Сами решения, как правило, являются простыми, и в отличие от решений другими методами не содержат утомительных вычислений. Это является очевидным достоинством графов, так как изобилие выкладок, как известно, не свидетельствует о содержательности теории. Теория графов притягательна как раз тем, что при всей своей наглядности и простоте помогает решать серьезные математические и прикладные проблемы.

Знакомство с отдельными разделами теории графов становится возможным уже в начальной школе при решении всевозможных логических задач и головоломок. Дальнейшее знакомство с графами в основной школе поможет при изучении многих математических разделов и будет служить хорошим подспорьем при решении сложных олимпиадных задач.

Происхождение теории графов

Существует большое количество практических задач, рассмотрение которых сводится к изучению совокупности объектов, существенные свойства которых описываются связями между ними. Например, на карте авиалиний интерес представляет лишь то, между какими городами имеется связь. При изучении электрических цепей на первый план выступает характер соединений различных ее элементов. Органические молекулы образуют структуры, характерными свойствами которых являются связи между атомами. Интерес могут представлять различные экономические связи, связи и отношения между людьми, событиями, состояниями, и вообще, между любыми объектами.

В подобных случаях удобно изображать рассматриваемые объекты точками, называя их *вершинами,* а связи между ними – линиями произвольной конфигурации), называя их *ребрами.*

Допустим, пять государств *A, B, C, D, E,* договориться о сотрудничестве, послали представителей на конференцию. Результатом конференции явилось подписание следующих договоров между *A* и *C, B* и *C, A* и *D*, *D* и *B, E* и *A, B* и *E.* Данной ситуации соответствует граф, имеющий пять вершин, шесть ребер.

Первая работа по графам была выполнена Л. Эйлером в 1736 г. и посвящалась решению знаменитой *задачи о кенигсбергских мостах*. предложенные Эйлером в этой работе, легли в основу теории уникурсальных графов. Наряду с решением этой чадами, Эйлер получил также ряд результатов, которые легли в основу проблемы планарности графов. В своих работах Эйлер не использовал термин «граф». Впервые этот термин в 1936 г. ввел Д. Кёниг, назвав графами схемы, состоящие из множества точек и связывающих эти точки отрезков прямых и кривых линий. Теория графов связана с именами многих известных математиков. Так, А. Кэли применил теорию графов к проблеме раскраски карты, а У. Гамильтон исследовал один интересный класс графов, названных впоследствии гамильтоновыми графами.

До конца XIX в. графы применялись лишь при решении некоторых занимательных задач. Однако в начале XX в. теория графов оформилась в виде самостоятельной математической дисциплины. Наряду с многочисленными головоломками и играми на графах появились важные практические приложения графов, многие из которых требовали тонких математических методов. Так, Кирхгоф применил графы для анализа электрических цепей, а Кэли исследовал важный класс графов для изучения насыщенных углеводородов.

В настоящее время теория графов широко применяется в различных областях науки и техники. К числу прикладных задач, решаемых при помощи графов, относятся, например, анализ и синтез цепей и систем, проектирование каналов связи и исследование процессов передачи информации, построение контактных схем и исследование конечных автоматов, сетевое планирование и управление, исследование математических операций, выбор оптимальных потоков в сетях, моделирование нервной системы живых организмов, исследование случайных процессов и многие другие задачи. Теория графов тесно связана с геометрией и топологией, теорией множеств и математической логикой, теорией вероятностей и математической статистикой, теорией матриц и другими разделами математики. Используя аппарат этих разделов математики, теория графов обогащает их методы и продолжает интенсивно развиваться.

Использование графов для решения задач информатики

Графы представляют собой наиболее общую структуру, с которой приходится работать при решении задач на ЭВМ. Графы используются для описания алгоритмов автоматического проектирования, в диаграммах машины конечных состояний, при решении задач маршрутизации потоков и т.д. Любая система, предполагающая наличие дискретных состояний или наличие узлов и переходов между ними может быть описана графом.

Графы широко используются для формирования структур данных в различных информационных системах. Основой информационной системы является база данных. Существуют следующие модели баз данных, которые используют графы:

- Графовая модель, данные размещаются в вершинах некоторого графа, а ребрами графа служат связи между данными (пример: Гипертекст).
- Иерархическая модель, данные размещаются в вершинах дерева. При этом дерево образует иерархию - каждая вершина соединяется только с вышележащей и нижележащими вершинами.
- Объектно-ориентированная модель, данные формируют объекты, которые размещаются в вершинах сети. Ребрами служат связи между объектами.

Для поиска информации в базах данных аналогично их структуре используется алгоритмы на графах - сортировки и поиска с помощью деревьев, обходов бинарных деревьев, поиска в глубину и ширину.

Многие важные алгоритмические задачи проще рассматривать с помощью графовой модели. Это задачи Прима-Краскала, Дейкстры, коммивояжера, нахождения максимального потока в сети, сетевого планирования, построения плоского графа при проектировании микросхем.

Понятие графа, методы представления графа

Понятие графа

Граф - это не пустое множество элементов (точки, вершины) связанных между собой определенными типами отношений.

∃ 3 условные определения Г.

1) **Топологическое**

   Граф изображается на плоскости с выделение и указанием вершин и ребер. Вершины обозначаются обычно однотипными фигурами (окружностями, треугольниками), а ребра отрезками прямых или кривых линий.

   Топологическое обозначение графа неоднозначно. Графы, у которых одинаковое количество вершин, ребер и ребер, соединяющих вершины между собой соответствующие вершины называются *изоморфными.* Изоморфные графы могут изображаться неодинаково, но с точки зрения теории графов это один и тот же граф.

2) **Алгебраическое**. Граф можно представить в виде алгебраической

системы, состоящей из двух множеств (вершин и ребер), а так же функции (ф-ии инцидентности).
Вершина и ребро считаются инцидентными, если вершина принадлежит данному ребру. Две вершины считаются смежными , если принадлежат одному и тому же ребру.

3) **Матричная форма.** Граф можно описать при помощи матрицы смежности или матрицы инцидентности.

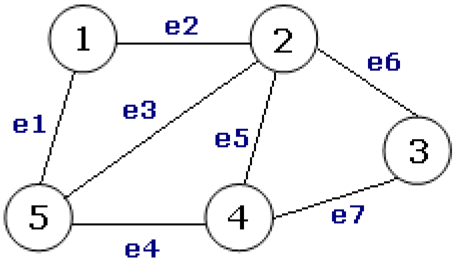
а) матрица смежности строится следующим образом: строится таблица число строк и столбцов, которой равно количеству вершин графа.
б) таблица заполняется по следующему принципу

Rij=1, если Vi и Vj – смежны.
Rij=0, если Vi и Vj - несмежны.

Где i-№ строки, j-№ столбца.

Вершина смежная самой себе называется петлей (в обычных задачах петли не встречаются).

Пример: пусть граф задан графически, списком смежных вершин, матрицей смежности и матрицей инцидентности.



список смежных вершин

номер вершины	**степень вершины**				
1	2	2	5		
2	4	1	3	4	5
3	2	2	4		
4	3	2	3	5	
5	3	1	2	4	

матрица смежности

	1	**2**	**3**	**4**	**5**
1	0	1	0	0	1
2	1	0	1	1	1
3	0	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

Если посчитать сумму цифр в строках, получим степень вершины Ci. Степень

вершины равна количеству ребер, инцидентных данной вершине.
матрица инцидентности

	e1	e2	e3	e4	e5	e6	e7
1	1	1	0	0	0	0	0
2	0	1	1	0	1	1	0
3	0	0	0	0	0	1	1
4	0	0	0	1	1	0	1
5	1	0	1	1	0	0	0

Если вычислить сумму в каждом столбце, то она будет равна 2. Это показывает , что у ребра только 2 вершины.
Таким образом, можно установить следующие свойства матриц:

- Сумма в каждой строке матриц равна степени вершины.
- Матрица смежности квадратная, а инцидентности в общем случае прямоугольная.
- Элементы главной диагонали матрицы смежности равны нулю.
- Относительно главной диагонали матрица смежности симметрична.
- Сумма в каждой столбце матрицы инцидентности равна 2.

Замечание: матрица смежности и инцидентности определяется из графа однозначно.

Свойства матриц графа

Различное представление графов определяется различными приложениями.

- Топологическое представление используется для решения простых задач, где решение можно найти простым наглядным способом.
- Алгебраическое представление используется в алгебре, когда графы используются как абстрактные объекты.
- Матричная форма используется для представления графа в ЭВМ.

Граф можно представить как совокупность двух множеств – множества вершин и множества ребер, связанных с вершинами (инцидентных вершинам). Таким образом, граф можно было бы описать в ЭВМ двумя связанными массивами, однако существуют более удобные варианты:

1. Матрица смежности. Квадратная матрица, в которой каждая строка и столбец соответствуют одной вершине графа. Если вершины связаны ребром, то на пересечении соответствующих строк и столбцов записывается число. Если связи нет, то записывается ноль. Есть вариант, когда записывают только число 1 (нет длин ребер) или число равное длине ребра. Матрица смежности симметрична относительно главной диагонали, на главной диагонали все значения равны нулю, количество ненулевых элементов строки или столбца равно степени соответствующей вершины.
2. Матрица инциндентности. Матрица в которой строки связаны с вершинами графа, а столбцы с ребрами графа. В общем случае не квад-

ратная. Как правило содержит нули и единицы. Ноль при отсутствии соединения вершины и ребра, 1 при наличии инциндентности. Главное свойство этой матрицы – число ненулевых элементов любого столбца равна 2. Количество ненулевых элементов строки равно степени соответствующей вершины.

3. Список ребер. Пронумерованный список пар вершин, которые связаны с ребром данного номера.

Наличие нескольких вариантов описания графа требует прежде всего построения алгоритмов проверки правильности введенных данных и возможности конвертации данных из одного варианта в другой.

Следующей задачей является визуализация графа. Так как любой граф можно изобразить на плоскости, то подобная задача имеет простой смысл: изображаем в случайных, несовпадающих точках вершины-круги и соединяем их отрезками, соответствующими ребрам графа.

Изоморфизм графов. Мультиграф, орграф, взвешенный граф

О. Различные изображения одного и того же графа называются изоморфными.

О. Изоморфизм — множественность представления объектов.

О. Мультиграф — граф, в котором есть кратные ребра. Если между одной парой вершин есть несколько ребер, то эти ребра называют кратными.

О. Если ребро соединяет одну и туже вершину, то оно называется петлей.

О. Граф в котором для ребер задается функция веса, называется взвешенным. Функция веса имеют аргументом ребра, а значением числовые величины, которые принято называть весами ребер.

Степень вершины графа

Определение 1 Вершины графа могут быть внутренними, если они принадлежат как минимум к 2-м ребрам, концевыми или висячими, если они принадлежат одному ребру и изолированными, если не принадлежат ни к одной.

Определение 2 По количеству вершин графы могут делиться на конечные и бесконечные. В дальнейшем мы будем рассматривать только конечные графы, не имеющие кратных ребер и петель.

Определение 3 Степенью вершины называется количество ребер инцидентных ей. По степени вершин графы делятся на графы с одинаковой степенью вершин – однородные и неоднородные. Принято так же считать четность и нечетность вершин в зависимости от четности их степени.

Определение 4 Полным графом принято называть граф, у которого вершина соединена с остальными. При заданном количестве вершин полный граф определяется однозначно. Степени всех вершин полного графа максимальны и одинаковы.

Определение 5 Пусть ∃ граф *G*. Дополнением к данному графу называется такой граф, вершины которого совпадают с G, а ребра дополняют этот граф до полного. Т.к. полный граф определяется однозначно по вершинам => по G определяется дополнение. Дополнением к полному графу называется граф, не имеющий ребер – ***нульграф.***

Свойства степеней вершин графа. Полный граф. Дополнение к графу

Рассмотрим некоторые важные для приложений свойства вершин графов:

Утверждение 1 Степень любой вершины полного графа из n вершин равна n-1, т.е. C = n-1.

Док-во: метод от противного, пусть степень одной из вершин C>n-1 или C<n-1, тогда вершина соединяется с n вершинами (чего быть не может) или не со всеми. Получили противоречие => наше предположение не верно, ч.т.д.

Утверждение 2 Задан граф G и количество ребер – p, количество вершин – n, тогда сумма всех степеней будет равна удвоенному количеству ребер.

Док-во: По определению C_i - количество ребер p_i, инцидентных вершине, вычислим Σ степеней всех вершин $\Sigma C_i = \Sigma p_i$. В эту сумму входят все ребра (так как нет ребер без вершин) и ровно 2 раза, так как любое ребро соединяется с 2 вершинами. Получили $\Sigma C_i = 2p$ ч.т.д.

Следствие: сумма степеней полного графа $\Sigma C_i = n*(n-1)$ => число ребер любого полного графа $p=n*(n-1)/2$

Утверждение 3 Число не четных вершин любого графа четно.

Док-во: По утверждению 2 сумма ΣC_i — четное число. Тогда число не четных вершин графа не может быть нечетно, так как сумма нечетного числа нечетных степеней это нечетное число, а сумма любого числа четных степеней это четное число. В результате сумма всех степеней станет нечетной, что противоречит утверждению 2. Получили, что число не четных вершин любого графа четно ч.т.д.

Утверждение 4 В любом графе из n-вершин, где n > 1, есть хотя бы одна пара вершин с одинаковыми степенями.

Док-во: метод от противного, пусть в графе с числом вершин n > 1 нет вершин с одинаковыми степенями. Расположим эти вершины в порядке убывания степеней. Максимальная степень может быть n-1, минимальная 0. Между этими числами ровно n мест (0,1,2,3...n-1). Так как нет одинаковых степеней, то все места заняты. Тогда есть вершина степени C=n-1 (связанная со всеми остальными) и вершина степени C=0 (изолированная), но этого не может быть.

Получили противоречие => наше предположение не верно, ч.т.д.

Утверждение 5 Задан граф из n-вершин, где $n \geq 2$, в котором 2 вершины имеют одинаковую степень, тогда в графе $\exists$ либо единственная изолированная вершина и нет максимальных степеней n-1, либо одна вершина максимальной степени n-1 и не одной изолированной.

Док-во: Метод от противного. Предположим при условии утверждения в графе существуют другие варианты степеней. Пусть задан граф с числом вершин $n > 1$, где только 2 вершины с одинаковыми степенями. При этом, в нем могут быть как изолированные вершины и вершины степени n-1. Очевидно, что в одном графе такие вершины быть не могут. Значит остаются варианты, когда есть более одной изолированной вершины и нет вершин степени n-1 либо есть более одной вершин степени n-1 и нет изолированных вершин. Расположим эти вершины в порядке убывания степеней. Максимальная степень может быть n-1, минимальная 0. Между этими числами ровно n мест (0,1,2,3...n-1). Так как есть только 2 вершины одинаковых степеней, то все места заняты кроме одного. Если есть изолированные вершины, то свободна максимальная степень и наоборот.

Рассмотрим 4 варианта:

1. Есть 2 изолированные вершины и все места заняты кроме степени n-1. Тогда есть максимальная степень n-2, это вершина которая связана со всеми кроме одной вершины, но этого не может быть (2 изолированые вершины). Получили противоречие.
2. Есть 2 вершины степени n-1 и все места заняты громе степени 0. Тогда есть минимальная степень 1, это вершина которая связана только с одной вершиной, но этого не может быть (2 вершины связаны со всеми). Получили противоречие.
3. Есть 2 вершины степени $0<m<n-1$ и все места заняты кроме степени 0 или n-1. Но это условие нашего утверждения.

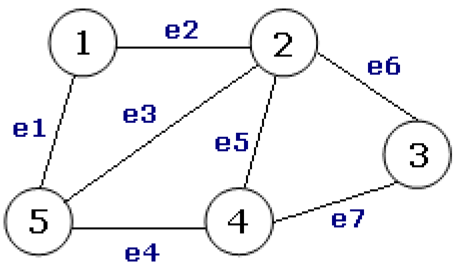
Таким образом, по всем вариантам предположения получили противоречие => наше предположение не верно, ч.т.д.

Цепь и путь в графе

__**Определение 6**__

Цепью или ***путем*** в графе называется последовательность ребер или вершин расположенных т.о., что ребра имеют общие вершины с соседними

Примеры:



1-5-2-3-цепь, а 2-4-1-5-не цепь. e1-e3-e2-e5-не цепь, а e2-e3-e4-e5- цепь.
Если цепь записывается в виде последовательности вершин, то соседние вершины должны иметь общее ребро. Цепь и путь, в принципе, одно и тоже, но понятие путь используется в ориентированных графах, а цепь в обычных случаях.

Определение 7

Цикл – это путь, у которого начальные и конечные вершины совпадают. В цикле могут повторятся ребра и вершины.

Определение 8

Простой цикл – цикл у которого нет повторяющихся вершин.
Пример: 1-2-3-4-5-1

Определение 9

Элементарный цикл– цикл у которого не повторяются ребра.
Пример: e1-e3-e6-e7-e5-e2 . В этом примере повторяются вершины (№2), но не ребра. Таким образом, элементарный цикл может быть не простым. В то же время простой цикл не может быть не элементарным (так как у одного и того же ребра и вершины одни и те же). Значит простых циклов в графе меньше.

Определение 10

Длинной пути или цепи называется число ребер данного пути или цепи.

Утверждение 6

Если в графе все простые циклы имеют четную длину, то не могут ∃ циклы нечетной длины.

Док-во: Метод от противного. Предположим при условии утверждения в графе существуют циклы нечетной длины. Пусть задан граф с простыми циклами четной длины и есть цикл нечетной длины. Значит этот цикл не простой. То есть у него повторяются вершины. Запишем этот цикл в виде последовательности вершин 1-3-4-...-1. Число ребер в цикле p (нечетное), число вершин в записи p+1 — четное число. При этом есть внутри номера вершин, которые повторяются. Пусть повторяется только вершина номер K и это не 1 (если это 1, то можно переписать цикл например 3-4-...-1-3). Разделим цикл на три части от 1 до первого K (I), от первого K до второго K (II) и от второго K до 1 (III). Очевидно, что все части не содержат повторений номеров и их можно переписать в виде 2 простых циклов (I)+(III) и (II). Поскольку они имеют четную длину, то и их сумма четна. Получили противоречие для повторения только 1-й вершины. Пусть повторяются 2 вершины, тогда по одной из этих вершин цикл можно разделить на 2 аналогичным способом, причем в каждом будет уже повторяться только 1 вершина. Снова получим противоречие. Далее этот метод можно расширить на 3 повтора и т. д. Поскольку число вершин конечно, то на каком-то этапе этот процесс остановится.Фактически это аналог метода матиндукции.

Таким образом, по всем вариантам предположения получили противоречие => наше предположение не верно, ч.т.д.

Определение 11

Подграфом графа называется такой граф, куда входят полностью часть вершин с прилежащими к ним ребрами.

<u>Определение 12</u>

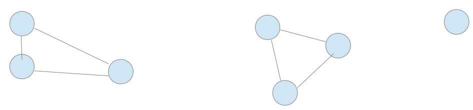
Вершины графа называются ***связными,*** если ∃ какой-нибудь путь между ними. Если все вершины графа связанны между собой, то граф называется ***связным.*** В связном графе, не может быть изолированных вершин.

Связность графа. Циклы графа

Граф, который не является связным, может быть представлен в виде совокупности нескольких связных графов отдельных друг от друга.

<u>Определение 13</u>

Компонентом связности графа называется подграф, который не имеет связи (пути) между вершинами этого подграфа и остальными вершинами графа.
Пример: данный граф имеет 3 компоненты связности



<u>Определение 14</u>

Мостом графа называется ребро после удаления которого теряется связность графа и образуется новая компонента связности. Мост не может входить в состав цикла графа, так как тогда существовал бы обходной путь вокруг моста. Фактически мост — пример антициклического ребра.

<u>Определение 15</u>

Ребро называется циклическим, если оно принадлежит хотя бы одному циклу графа. Ребра, которые не являются циклическими принято называть антициклическими. Очевидно, что антициклическое ребро должно быть мостом, так его удаление как минимум разрывает связь между вершинами этого ребра.

Деревья. Свойства деревьев

<u>Определение 16</u>

Граф называется **деревом**, если он связан и не имеет циклов.

Утверждение 6

Граф является деревом только тогда, когда каждая пара различных вершин соединяется одной и только одной цепью.
Док-во : метод от противного, пусть это условие выполнено для графа, который не является деревом. Тогда существуют циклы в этом графе => между вершинами цикла есть 2 разных пути их соединяющих => противоречие => ч.т.д.

<u>Определение 17</u>

Если все вершины графа G совпадают с вершинами дерева T и все ребра дерева T являются ребрами графа G, то дерево T называетя ***остовным*** для графа G.

Утверждение 7
Дерево с n вершинами имеет ровно n-1 ребро.
Док-во : метод от противного, пусть в дереве число ребер >=n. Представим процесс удаления из этого графа висячих вершин. Если в графе нет висячих вершин, то они все имеют хотя бы 2 смежных ребра => можно строить путь через вершины так, чтобы можно было приходить в вершину и уходить из вершины. В конце такого пути будет или висячая вершина или образуется цикл. Но это не возможно в дереве => обязательно есть висячая вершина. Удалим ее и ребро, связывающее ее с деревом. Число вершин уменьшается на 1 и число ребер уменьшается на 1. В конце концов остается 1 ребро и 2 его вершины => в дереве число ребер на 1 меньше числа вершин. Ч.Т.Д.
Следствие 1: У любого дерева ∃ хотя бы одна висячая вершина.
Следствие 2: Если к дереву добавить висячую вершину, то дерево останется деревом.
Следствие 3. Дерево можно считать минимальным по числу ребер связным графом данного набора вершин.
Утверждение 8
Любой граф, который не является деревом имеет хотя бы один простой цикл.
Док-во : метод от противного, пусть в графе есть составной цикл, но нет простого цикла, где не повторяются вершины. Однако любой составной цикл можно разделить по вершине которая повторяется.
Пусть 1-2-3-4-......-4-5-6-7-1 такой цикл, тогда 4-2-3-1-7-5-6 — простой цикл. И так можно поступить всегда. Ч.Т.Д.
Утверждение 9
Удаление из графа циклического ребра не меняет его связности, а антициклического меняет.
Док-во: Во-первых, удаление из графа циклического ребра не меняет его связности так как в цикле всегда есть 2 разных пути между парой вершин => связь остается. Во-вторых, удаление из графа антициклического ребра меняет его связность так как такого обходного пути нет, иначе был бы цикл и ребро было бы циклическим. Ч.Т.Д.

Следствие. Если в графе с n - вершинами ровно n-1 ребро и есть связность вершин, то это дерево.

Остовное дерево графа

Утверждение 10
Из любого графа можно получить дерево путем удаления циклических ребер.
Дерево получаемое из графа путем удаления циклических ребер будет ***остовным (доказательство очевидно).***

Утверждение 11
В любом дереве существует только 1 путь между любыми 2-мя вершинами дерева ***(доказательство очевидно из утверждения 6).***

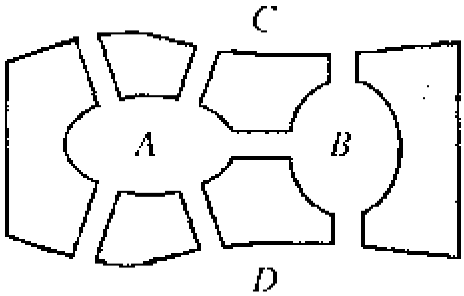
Лекция №4

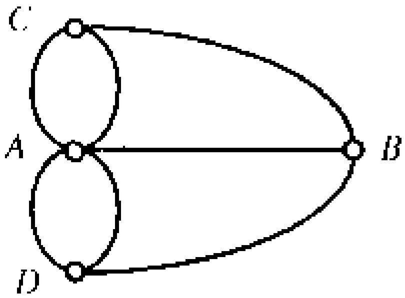
Раздел №2. Эйлеровы, и Гамильтоновы графы

Содержание: Задача о Кенигсберских мостах. Эйлеров цикл. Уникурсальная линия. Теорема Эйлера. Алгоритм построения Эйлерова цикла.

1.Задача Эйлера о кенигсбергских мостах

Основоположником теории графов считается Л. Эйлер. В 1736 г. им была выполнена работа, в котором содержалось решение знаменитой *задачи о **кенигсбергскш** мостах.*





На рисунке схематично изображена карта г. Кенигсберга XVIII века. Город был расположен на берегах и двух островах реки Преголи. Острова между собой и между берегами были связаны семью мостами. Возник вопрос: можно ли пройти по всем мостам так, чтобы на каждом из них побывать лишь один раз и вернуться к тому месту, откуда началась прогулка?

Идеи Эйлера, использованные им при решении этой задачи, явились фундаментом теории, впоследствии названной *теорией графов.*

Ясно, что по условию задачи не имеет значения, как проходит путь по частям суши A, B, C, D, поэтому их можно изобразить точками. А так как связи между этими частями суши осуществляются только через семь мостов, то каждый из мостов можно изобразить линией, соединяющей соответствующие вершины. В результате получается граф. Если бы существовал маршрут движения, удовлетворяющий условию задачи, то этот граф было бы возможно нарисовать *«одним росчерком»* (т.е. *без отрыва карандаша от бумаги, проводя по каждому ребру только один раз),* начиная и заканчивая рисование в одной точке. Эйлером было доказано, что это невозможно. Возникает вопрос: будет ли задача о кенигсбергских мостах иметь решение, если отказаться от того, чтобы маршрут движения начинался и заканчивался в одной точке? В этом случае мы вновь приходим к задаче об изображении графа одним росчерком. Как будет показано ниже, одним росчерком изобразить невозможно.

Определение 18. Эйлеровым циклом называется элементарный цикл, содержащий все ребра.

Определение 19. Граф *G* называется *эйлеровым.,* если он содержит эйлеров цикл.

Определение 20. Уникурсальной линией (уникурсальной цепью или эйлеровым путем) называют элементарную цепь, проходящую по всем ребрам графа, но не являющуюся циклом.

<u>Определение 21</u>. Эйлеров путь (ЭП)- путь содержащий все ребра графа без повторений (элементарный путь).

Теорема Эйлера

Необходимым и достаточным условием существования Эйлерова цикла является четность всех степеней вершин графа.

Док-во:
Необходимость. Предположим, что *C* является эйлеровым циклом в графе G. Для произвольной вершины графа цикл *C* входит в нее по одному ребру и выходит по другому. Поскольку каждое ребро представлено в цикле ровно один раз, то каждая вершина должна иметь четную степень.

Достаточность.

1. База индукции очевидна. m=1 – петля, m=2 – кратные ребра, m=3 – рисунок.

2. Предположим, что связный граф, имеющий только четные вершины и содержащий $n=k$ ребер, является эйлеровым.

3. Рассмотрим связный граф G, имеющий только четные вершины и содержащий $n = k+ 1$ ребро. Так как граф связен, он не имеет изолированных вершин. Пусть A_0 -некоторая вершина графа. Так как A_0 - не изолированная вершина четной степени, то существует ребро e_1, исходящее из A_0, и мы можем по этому ребру перейти в некоторую вершину A_2. Вершина A_2; имеет четную степень, следовательно существует ребро e_2, отличное от e_1 и выходящее из A_2. Переходя по ребру e_2, мы попадем в вершину A_3, и так далее... После конечного числа таких переходов мы вновь вернемся в вершину A_0, т.е. пройдем некоторый цикл с вершиной A_0. Если пройденный цикл включает в себя все ребра графа, то он - искомый цикл. Предположим, что пройденный цикл не включает всех ребер графа. Удалим из исходного графа ребра выделенного цикла. В результате получим подграф, который имеет несколько компонент связности (возможно, всего одну) G_i *(1=1, 2, ... ,r).* При этом будут выполняться следующие свойства:

- каждая часть G_i является связным графом;
- каждая часть G_i, имеет лишь вершины четной степени;
- каждая часть G, хотя бы одной вершиной примыкает к вершинам удаленного цикла;
- каждая часть G,- имеет не более k ребер.

По предположению индукции каждая из частей G, имеет эйлеров цикл, причем начало и конец пути могут находиться в любой из вершин (в частности ими могут быть точки примыкания к вершинам удаленного цикла). Итак, алгоритм построения общего эйлерова цикла можно сформулировать так: двигаясь из вершины A_0 по выделенному циклу, попутно переходить в указанные выше связные части G, проходить их внутренний эйлеров цикл и возвращаться в исходный выделенный цикл. После завершения выделенного цикла будет завершен и цикл по всему графу. Таким образом, построение Эйлерова цикла в графе содержащем $n = k+ 1$ ребро возможно. По методу математической индукции делаем вывод, что теорема верна для графов с любым конечным количеством ребер. ч.т.д.

Алгоритм построения Эйлерова цикла

1. Считаем степень вершин и проверяем их на четность;
2. Выделяем все простые циклы и удаляем их;
3. Выделяем общие вершины;
4. Составляем цепь из простых циклов, так чтобы проходя их один за другим можно было бы последовательно пройти по всем найденным простым циклам.
5. По выделенной цепи циклов строим общий Эйлеров цикл графа.

Следствие: для того чтобы Э.графе ∃ уникурсальная линия между вершинами А и В, необходимо чтобы А и В были единственными нечетными вершинами графа.

Пример : Полный граф из 5 вершин имеет все вершины степени 4 и поэтому является Эйлеровым. Обозначим 1,2,3,4,5 вершины графа и построим Эйлеров цикл без графических построений.

Шаг 1. Составим список ребер графа, их должно быть р=5*4/2=10. Все соединяются со всеми, поэтому в порядке возрастания:

ребра	р1	р2	р3	р4	р5	р6	р7	р8	р9	р10
а_в	1_2	1_3	1_4	1_5	2_3	2_4	2_5	3_4	3_5	4_5

Шаг 2. Последовательно выделяем простые циклы и удаляем из таблицы:

Цикл а) 1_2_3_4_5_1. Остались 5 ребер.

ребра	р1	р2	р3	р4	р5	р6	р7	р8	р9	р10
а_в		1_3	1_4			2_4	2_5		3_5	

Цикл с) 1_3_5_2_4_1. Ребер не осталось.

Шаг 3. Выделим у циклов а) и с) общую вершину, например 4. Теперь проходим цикл а) до 4, затем переходим в с) проходим его по кругу, через 4 возвращемся в а) и завершаем его.

Итого 1_2_3_4+переход 4_1_3_5_2_4+возврат+ 4_5_1

Ответ: Эйлеров цикл 1_2_3_4_1_3_5_2_4_5_1

Теорема Эйлера обобщенная

Если связный граф имеет 2к вершин, то в нем можно найти к различных

путей, которые содержат все ребра графа точно по 1-му разу.
Следствие 1. Эйлеров цикл всегда можно построить из нескольких простых циклов, соединенных общими вершинами и не имеющими общих ребер.

Следствие 2. Любое дерево должно иметь хотя бы одну нечетную вершину.

Теорема. *Связный граф является эйлеровой цепью тогда и только тогда, когда он имеет ровно две нечетные вершины, а остальные вершины этого графа четны. При этом начало и конец уникурсального пути находятся в нечетных вершинах.*

Доказательство. Обозначим через *A* и *B* нечетные вершины графа G. Так как граф связен, то между этими вершинами существует простая цепь. Если эта цепь включает все ребра графа, то она является искомым уникурсальным путем. Предположим, что выделенная цепь не включает всех ребер графа. Удалим ребра выделенной цепи. В результате получим подграф, который имеет несколько компонент связности (возможно, всего одну) G_i *(1=1, 2, ... ,r)*.. При этом будут выполняться следующие свойства:

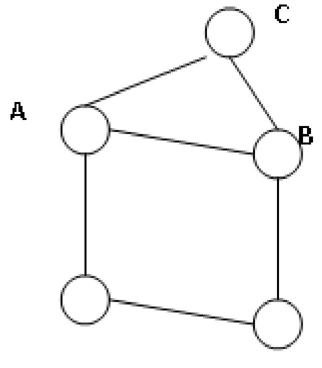
- каждая часть G, является связным графом;
- каждая часть G, имеет лишь вершины четной степени;
- каждая часть G, хотя бы одной вершиной примыкает к вершинам удаленной цепи.

По теореме каждая из частей G_i, является эйлеровым циклом, причем начало и конец пути могут находиться в любой из вершин (в частности ими могут быть точки примыкания к вершинам удаленной цепи). Следовательно, алгоритм построения уникурсалыюй цепи можно сформулировать так: двигаясь из вершины *A* по выделенной цепи к вершине *B* попутно обрисовывать выделенные циклы G, (рисунок). Теорема доказана.

Следствие. Если граф имеет только одну нечетную вершину, то он не является полуэйлеровым.

Доказательство. От противного.

Пусть в данном графе существует уникурсальная линия, которая начинается в вершине А и заканчивается в вершине В. Замкнем уникурсальную линию, получив таким образом эйлеров цикл. Возможны два варианта:



1. Вершины A и B не имеют общего ребра. Добавим это ребро. Такое добавление превращает уникурсальную линию в эйлеров цикл, следовательно, добавив одно ребро, мы получили эйлеров граф. По теореме все вершины эйлерова графа – четны, однако, в нашем графе все вершины не могут иметь четные степени. Пришли к противоречию, значит, наше предположение не верно.
2. Вершины A и B смежны. Тогда можно добавить вершину C , соединив её с A и B. Такое добавление превращает уникурсальную линию в эйлеров цикл, то есть полученный граф – эйлеров. По теореме в нем не может быть нечетных вершин, однако, этого не может быть (по крайней мере одна вершина имеет нечетную степень). Вновь пришли к противоречию. Следовательно, граф, имеющий одну нечетную вершину не является полуэйлеровым.

Лекция №5

Раздел №2. Эйлеровы и Гамильтоновы графы

Содержание: Гамильнонов путь и цикл. Задача Гамильтона. Признаки существования Гамильтонова пути и цикла в графе. Задача коммивояжера. Путь и цепь в орграфе.

1. Гамильнонов путь и цикл

При решении проблемы уникурсальности графов нас интересовало существование в графе элементарных цепей или циклов, проходящих через все его ребра. Естественно попытаться решить похожую задачу о существовании простых цепей и циклов, проходящих через все вершины графа. Другими словами, можно ли, проходя по ребрам графа, обойти все его вершины, побывав и каждой из них ровно один раз?

Определение 22. *Простой цикл, проходящий через все вершины графа, называется **гамильтоновым циклом**, а простая цепь, включающая все вершины графа называется **гамильтоновой цепью**.*

Это связано с тем, что впервые такая задача была сформулирована У. Гамильтоном в 1859 г. в виде головоломки на додекаэдре. В ней требовалось указать замкнутый маршрут из ребер додекаэдра, проходящий по каждой вершине ровно один раз.

Определение 23. Конечный граф называется *гамилътоновым,* если он обладает гамильтоновым циклом, либо полугамильтоновым если есть гамильтонова цепь.

2.Признаки существования Гамильтонова пути и цикла в графе

Ясно, что для каждого графа, для которого существует гамильтонов цикл, можно указать также и гамильтонову цепь. Для этого достаточно разорвать указанный цикл, удалив из него одно ребро (например последнее в цикле). Обратное, вообще говоря, не верно. Так, например, дерево, являющееся простой цепью, вообще не имеет циклов. Далеко не каждый граф содержит гамильтонову цепь (и тем более гамильтонов цикл). К сожалению, до сих пор не получено необходимого и достаточного условия гамильтоновости графов. Более того, не существует хорошего алгоритма (не сводящегося к полному перебору путей) для оиискания гамильтоновых цепей в графах общего вида. И теории гамильтоновых графов известно лишь несколько интересных результатов для некоторых классов графов специального вида.

Некоторые свойства гамильтоновых графов

Как и для проблемы эйлеровости, связность графов является необходимым (но не достаточным) условием их гамильтоновости. Для некоторых классов связных графов гамильтоновость легко может быть доказана. Приведем некоторые достаточные признаки гамильтоновости для элементарных графов.

Утверждение 12

Связный граф с числом вершин n>=3 обладает гамильтоновым циклом , если для любой вершины A степень p(A)≥ n/2

Теорема. *Пусть связный граф имеет n вершин и* $n \geq 3$*. Если в этом графе для степени любой вершины A выполняется неравенство* $\sigma(A) \geq \dfrac{n}{2}$*, то он обладает гамильтоновым циклом.*

Доказательство. К графу G добавим *k* новых вершин и каждую из них соединим с каждой вершиной графа *G*. Получим новый граф *G'*. Ясно, что если выбрать *k* достаточно большим, то граф *G'* будет гамильтоновым. Предположим теперь, что *k* - минимальное из чисел, при котором в графе *G'* появляется гамильтонов цикл. Наша цель состоит в том, чтобы доказать, что *k=0.*

Предположим противное, что *k>0,* и пусть *P* - одна из добавленных вершин. Рассмотрим гамильтонов цикл в графе *G'* (он существует по предположению и содержит точку P).

Рядом с вершиной *P* могут находиться только вершины из графа G, обозначим их через *A и B*.

Вершины *A* и *B* не могут быть смежны, так как в противном случае мы могли бы не использовать вершину P, а это противоречит предположению о минимальности *k.* Так как по условию теоремы степень каждой вершины *A* больше 1, то в гамильтоновом цикле найдется вершина *A'* из G, смежная *A.* Обозначим через *C* вершину, непосредственно следующую за *A'.* Докажем, что вершины *B и C* не смежны. Действительно, если предположить противное, то наряду с рассматриваемым гамильтоновым циклом будет существовать еще один гамильтонов цикл, в котором участок от *B* до *A'* обращен в противоположное направление и не участвует точка *P.* Т.е. мы опять приходим в противоречие с предположением о минимальности *k.*

Итак, за точкой *A',* смежной вершине *A,* непосредственно следует вершина C, обязательно не смежная вершине *B.* Т.е. в графе *G'* число вершин, не смежных *B,* не меньше числа вершин, смежных *A*. Следовательно, число вершин в *G'*, не смежных *B,* равно по меньшей мере $\frac{n}{2}+k$.С другой стороны, число вершин в G', смежных *B,* тоже равно, по меньшей мере, . А так как одна и та же вершина не может DM быть одновременно смежной и не смежной вершине /7, то общее число вершин графа *G'* не меньше *n + 2k* . Но граф *C!'* имеет всего *n + k* вершин. Из равенства *n + 2k = n + k* следует противоречие с предположением о том, что *k>0.* Итак *k=0,* и граф G обладает гамильтоновым циклом. Теорема доказана.

Утверждение 13

Пусть задан граф с числом вершин n≥ 3, если для ∀ пары вершин A и B выполняется правило *p(A)+p(B ≥ n*, то этот граф является гамильтоновым и есть гамильтонов цикл.

Пусть элементарный связный граф имеет n вершин и n≥3. Если в этом графе для любой пары вершин A и B выполняется неравенство p(A)+p(B) ≥ n, то он обладает гамильтоновым циклом.

Доказательство:

Метод мат. индукции.

1. *n*=3 – выполняется.
2. Индуктивное предположение. Пусть условие выполняется для графа с количеством вершин, равным *k*.
3. Докажем утверждение для *n=k+1*. Итак, мы имеем граф с количеством вершин *k*, для которого выполняются условия доказываемого утверждения, то есть для любых двух вершин $\sigma(A)+\sigma(B)\geq n$, и в этом графе существует гамильтонов цикл. Граф с количеством вершин *k*+1 получается добавлением одной вершины к рассмотренному графу. Степень вершины *X*, которую мы добавляем должна быть больше 1 (иначе не выполнится условие следствия), следовательно, $\sigma(X)\geq 2$. Пусть степень этой вершины *X* $\sigma(X)=m\geq 2$. Построим гамильтонов цикл для предшествующего графа *k* вершин. Поскольку степень вершины $\sigma(X)\geq 2$, то она соединяется с двумя и более вершинами. Если степень 2, то должна существовать вершина, у которой степень *k-1*. Условие $\sigma(A)+\sigma(B)\geq n$, *n=k+1*, должно выполняться для всех вершин графа, тогда все вершины должны иметь степень *k-1*, то есть предшествующий граф был полный. В полном графе, по **следствию 1**, существует гамильтонов цикл, а, следовательно, этот цикл проходит и через вершины, с которыми связана новая вершина. Таким образом, если мы начнем цикл в вершине *X*, то мы точно можем его построить.

Если *m*>2, то степени вершин могут быть меньше *k*-1, а именно *k*-1, *k*-2. В этом случае у нас появляется выбор. Третья вершина, с которой соединена *X* может попарно встречаться в разных вариантах. Наихудший случай, когда *k*-2. Это

значит, что вершины соединяются со всеми, кроме одной. Выберем какую-либо пару вершин, с которой соединена X и возьмем это за начало гамильтонова цикла. Остальная часть графа без этих трех вершин, то есть выбранной пары и X, удовлетворяет условию и должна иметь гамильтонов цикл. Найдем в цикле две соседние вершины, связанные с этими удаленными вершинами.
Для любых двух соседних вершин из трех вершин мы всегда можем найти 2 связанные с ними. Поэтому мы можем построить гамильтонов цикл и через эти вершины.
Такие же рассуждения можно провести и для m=4 и так далее. При достижении $m=\frac{k}{2}$ начинает действовать условие теоремы => граф, полученный путем добавления одной вершины к графу k вершин и для которого выполняется условие $\sigma(A)+\sigma(B)\geq n$ гамильтонов. **Чтд**.

Теорема. *Если при n≥3 элементарный связный граф имеет n вершин и для любой его пары вершин A и B выполняется неравенство p(A)+p(B)* ≥ *n*–1, *то он обладает гамильтоновой цепью.*

Доказательство. Добавим вершину P к графу G и соединим ее ребрами со всеми другими вершинами. В результате получим граф G' имеющий n +1 вершину. Степени вершин (всех, кроме вершины P) увеличились на 1, поэтому для любых двух вершин A и B (отличных от P) справедливо неравенство $\sigma(A)+\sigma(B)\geq(n-1)+1+1=n+1$. С другой стороны, $\sigma(P)=n$, а $\sigma(A)\geq 1$ для любой вершины A. Следовательно $\sigma(A)+\sigma(P)\geq n+1$. Итак для любой пары вершин A и B графа G' имеет место неравенство $\sigma(A)+\sigma(B)\geq n+1$. Поэтому, в силу следствия из теоремы, граф G' обладает гамильтоновым циклом. Удалив из этого цикла вершину P вместе с парой примыкающих к ней ребер, получим гамильтонову цепь в графе G. Теорема доказана.
Ясно, что в элементарном связном графе, имеющем всего две вершины (в этом случае граф является отрезком), всегда существует гамильтонова цепь (это ребро, связывающее концы отрезка), но не существует гамильтонова цикла.

Утверждение 14

В полном графе:
а) количество различных гамильтоновых цепей n!
б) любая неповторяющаяся последовательность вершин является гамильтоновым путем.

Доказательство:
1. Так как вершина соединена со всеми остальными, то в качестве начальной можно выбрать любую вершину, в качестве второй – любую из оставшихся, и так далее.
2. Так же строим гамильтонов путь. Так как любая вершина соединена со всеми стальными, то, очевидно, любая последовательность вершин - является цепью, так как вершины не повторяются, то это – гамильтонов путь.

Если данный путь содержит все вершины, то мы получим гамильтонов цикл. Подсчитаем количество различных путей. Это есть число перестановок из *n* элементов, равное *n*!

3.Задача коммивояжера (ЗК)

Задача коммивояжера (в дальнейшем сокращённо - ЗК) является одной из знаменитых задач теории комбинаторики. Она была поставлена в 1934 году, и об неё, как об Великую теорему Ферма обламывали зубы лучшие математики. В своей области (оптимизации дискретных задач) ЗК служит своеобразным полигоном, на котором испытываются всё новые методы. Она относится к классу наиболее сложных задач (не полиномиальных NP).

Задача о коммивояжере: торговец должен посетить по 1 разу все вершины (города) графа, так чтобы вернуться в исходный город и проделать путь минимальной длины. Длины ребер графа заданы (граф взвешенный).

Условия задачи коммивояжера

Коммивояжер (переезжающий торговец) хочет посетить n городов, т. о. чтобы побывать в каждом только один раз, но побывать во всех городах и проехать минимальное расстояние. Предполагается, что известно расстояние между $\forall$ парой городов. Фактически задача о коммивояжере - это задача о поиске гамильтонова цикла (цикл по всем вершинам без повторений) кратчайшей длины. В общем случае эта задача может быть решена только перебором значений, но есть метод который позволяет упростить перебор отбрасыванием больших наборов вариантов. Этот метод называется методом ветвей и границ.

Метод ветвей и границ – это метод отбрасывания при полном переборе больших множеств вариантов, которые заведомо не содержат искомого оптимального решения. Например длина циклов в любом из вариантов этого множества больше чем в каком-то опорном цикле. Опорный цикл при этом может быть получен каким-либо неточным алгоритмом (жадным или деревянным).

Жадный алгоритм

Жадный алгоритм – алгоритм нахождения наикратчайшего расстояния путём выбора самого короткого, ещё не выбранного ребра, при условии, что оно не образует цикла с уже выбранными рёбрами. «Жадным» этот алгоритм назван потому, что на последних шагах приходится жестоко расплачиваться за жадность.

Деревянный алгоритм.

Теперь можно обсудить алгоритм решения ЗК через построение кратчайшего остовного дерева. Для краткости будет называть этот алгоритм деревянным.

Построим на входной сети ЗК кратчайшее остовное дерево и удвоим все его ребра. Получим граф G – связный и с вершинами, имеющими только четные степени.

Построим эйлеров цикл G, начиная с вершины 1, цикл задается перечнем вершин.

Просмотрим перечень вершин, начиная с 1, и будем зачеркивать каждую вершину, которая повторяет уже встреченную в последовательности. Останется цикл, который и является результатом алгоритма.

Пример 1. Дана граф заданный матрицей смежности. Найти тур жадным и деревянным алгоритмами.

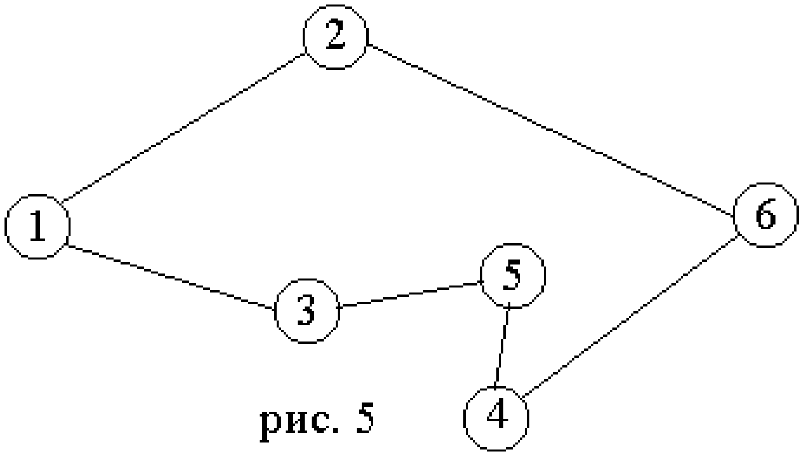


рис. 5

-	1	2	3	4	5	6
1	-	6	4	8	7	14
2	6	-	7	11	7	10
3	4	7	-	4	3	10
4	8	11	4	-	5	11
5	7	7	3	5	-	7
6	14	10	10	11	7	-

табл. 1

Решение. Жадный алгоритм (иди в ближайший город из города 1) дает цикл 1–(4)–3-(3)–5(5)–4–(11)–6–(10)–2–(6)–1, где без скобок показаны номера вершин, а в скобках – длины ребер. Длина цикла равна 39, цикл показана на рис. 5.

2. Деревянный алгоритм вначале строит остовное дерево, показанное на рис. 6 штриховой линией, затем эйлеров цикл 1-2-1-3-4-3-5-6-5-3-1, затем цикл 1-2-3-4-5-6-1 длиной 43, который показан сплошной линией на рис. 6.

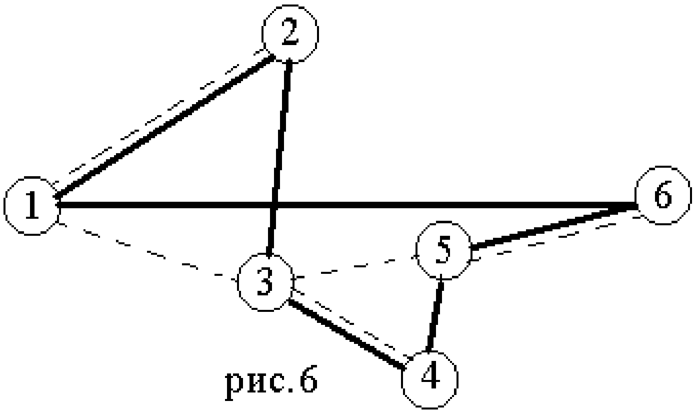


рис. 6

Таким образом, оба алгоритма дают некий начальный цикл, длина которого ограничивает перебор новых циклов.

Полный перебор практически применим только в задачах малого размера. Напомним, что ЗК с n городами требует при полном переборе полного графа рассмотрения n! циклов! Число таких варинтов, растет очень быстро:

5!	10!	15!	20!	25!
120	3628800	$1{,}3*10^{13}$	$2{,}4*10^{18}$	$1{,}5*10^{25}$

Поэтому используя любой метод, который дает возможность сократить перебор мы улучшаем время решения этой задачи.

Лекция №6

Раздел №3. Прикладные задачи теории графов

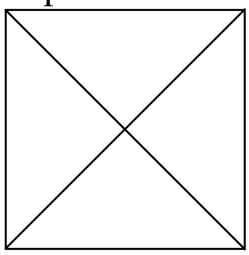
Содержание: Плоские графы. Эйлерова характеристика графа. Теорема Понтрягина-Куратовского. Построение плоского графа. Задача трассировки.

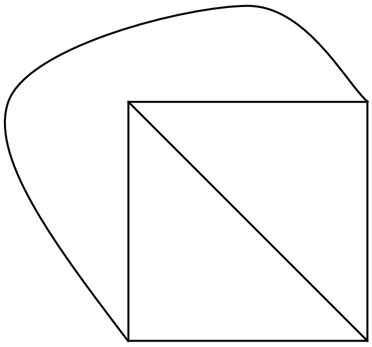
1. Плоские и планарные графы

Для представления графов мы пользовались диаграммами, на которых точки или кружки изображали вершины, а прямолинейные отрезки или дуги кривых — ребра. Такие диаграммы очень удобны при исследовании свойств отдельных графов. Естественно поэтому спросить, что это на самом деле означает — «представить» граф с помощью диаграммы, и всякий ли граф может быть представлен таким образом?

Было бы хорошо, если бы мы имели возможность изображать графы в некотором пространстве (например, на плоскости или в трехмерном евклидовом пространстве) так, чтобы они не имели «пересечений» (формальное определение этого понятия будет дано позднее, а его интуитивное значение очевидно). Например, изображение графа K_4

содержит пересечение; мы же хотим найти такое изображение, которое не содержит пересечений.





В действительности каждый граф может быть изображен без пересечений в трехмерном пространстве, но, как мы увидим, на плоскости такое представление возможно не всегда. В частности, графы K_5 и $K_{3,3}$ не могут быть изображены на плоскости без пересечений.

Говорят, что граф G **может быть уложен** (или **обладает укладкой)** в данном пространстве, если он изоморфен некоторому графу, изображенному в этом пространстве при помощи точек, представляющих вершины G, и кривых, не имеющих самопересечений и не пересекающихся между собой.

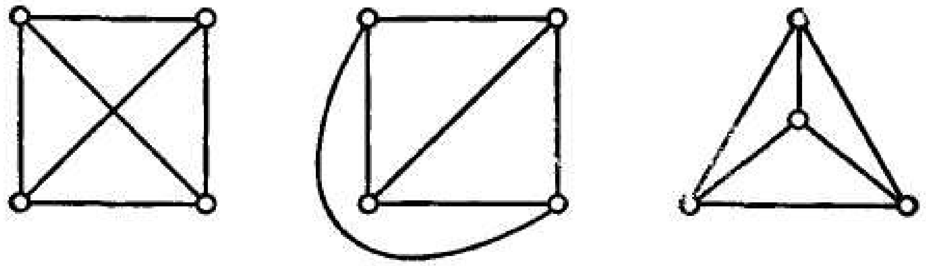
Теорема. *Каждый граф может быть уложен в трехмерном евклидовом пространстве.*

Эта теорема дает нам искомое обоснование использования диаграмм для изображения графов: достаточно взять трехмерное представление и спроектировать его на плоскость так, чтобы никакие две вершины не попадали в одну и ту же точку. Конечно, в общем случае такой метод приводит к пересечениям, хотя в некоторых случаях мы получим диаграммы без пересечений. Произойдет это только тогда, когда рассматриваемый граф может быть уложен на плоскости; такие графы называются **планарными графами.**

Теперь мы приступаем к изучению топологической теории графов. Здесь элементы теории графов тесно переплетаются с такими топологическими понятиями, как планарность. Было доказано, что каждый граф может быть уложен (т. е. изображен без пересечений) в трехмерном пространстве; теперь мы исследуем условия, при которых граф можно уложить на плоскости и на других поверхностях.

О. Плоским графом называется граф, изображенный на плоскости так, что никакие два его ребра (или, вернее, представляющие их кривые) геометрически не пересекаются нигде, кроме инцидентной им обоим вершины.

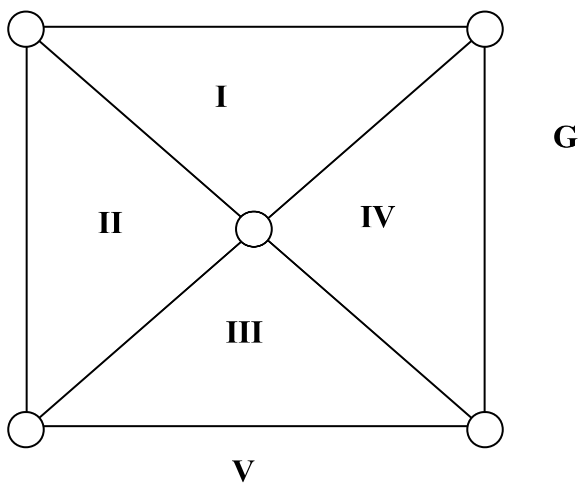
О. Граф, изоморфный плоскому графу, называется **планарным. М**ожно сказать, что граф планарен, если его можно уложить на плоскости. Например, все три графа на рис. планарны, но только второй и третий из них плоские. Глядя на этот пример, естественно поставить следующий вопрос: всегда ли можно изобразить планарный граф на плоскости так, чтобы все его ребра были представлены прямолинейными отрезками? Хотя очевидно, что для графов, содержащих петли и кратные ребра,



это не верно, для простых графов это действительно так, что было доказано Вагнером в 1937 г.

2. Эйлерова характеристика графа

О. *Гранью* называется часть графа, ограниченная простым циклом и не имеющая внутри себя других простых циклов.



Граф G, например, имеет 5 граней, одна из них – внешняя.

О. Эйлерова характеристика графа S = n – p + f где f – число граней

Теорема о Эйлеровой характеристике (Эйлер 1752). *Пусть G — связный обыкновенный плоский граф, пусть **n**, **p** и **f** обозначают соответственно число вершин, ребер и граней графа G. Тогда S = 2*

Доказательство. Доказательство проводится индукцией по числу вершин в *G*. Если n=2, то p=1 (так как *G* связен) и *f=1* (бесконечная грань); итак, в этом случае теорема верна.

Предположим теперь, что теорема верна для любого графа *G*, имеющего n вершин, и добавим к *G* новую вершину. Тогда эта вершина либо принадлежит внутренней грани, либо принадлежит внешней грани в плоской укладке этого графа.

Пусть вершина находится на внешней грани, тогда есть несколько ребер, которые соединяют ее и вершины, которые находятся на внешней границе остального графа (иначе это не будет плоская укладка). Вновь метод индукции. Если 1 ребро, то новых граней нет, +1 ребро +1 вершина => Эйлерова характеристика не изменится. При добавлении каждого нового ребра +1 грань и +1 ребро => Эйлерова характеристика не изменится. Значит для такого графа Эйлерова характеристика не изменится.

Пусть вершина находится внутри внутренней грани, тогда есть несколько вершин этой грани, которые образуют простой цикл. Новая вершина может иметь общие ребра, которые соединяют ее и вершины этого цикла. С другими вершинами графа таких ребер нет (иначе это не плоская укладка). Теперь рассуждение аналогичное как для предшествующего случая. Первое ребро не дает новых граней, а второе, третье и последующие добавляют новые грани => Эйлерова характеристика не изменится.

Таким образом, по методу математической индукции любой плоский граф имеет *S* = *2, ч.т.д.*

Теорема доказана.

Теорему Эйлера легко перенести на несвязные графы:

Следствие 1. *Пусть G — плоский граф с* ***n*** *вершинами,* ***p*** *ребрами,* ***f*** *гранями и* ***k*** *компонентами; тогда*

$$\boldsymbol{n+f=m+k+1}$$

Доказательство. Результат получается непосредственным применением теоремы Эйлера к каждой компоненте по отдельности. При этом бесконечная грань считается только один раз. //

Все упомянутые до сих пор результаты этого параграфа применимы к произвольным плоским графам; далее ограничимся простыми графами.

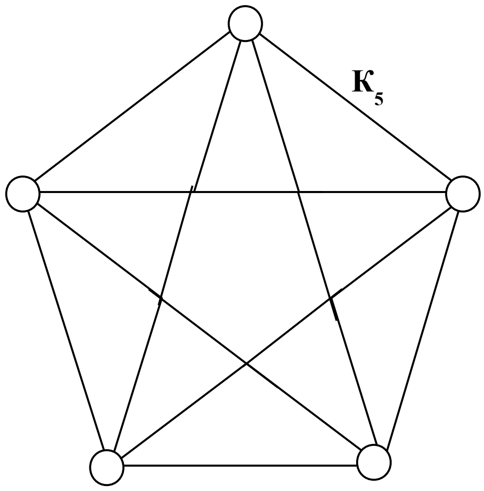
Следствие 2. *Если G — связный простой планарный граф с* $\boldsymbol{n \geq 3}$ *вершинами и* ***p*** *ребрами, то* $\boldsymbol{p \leq 3n-6}$

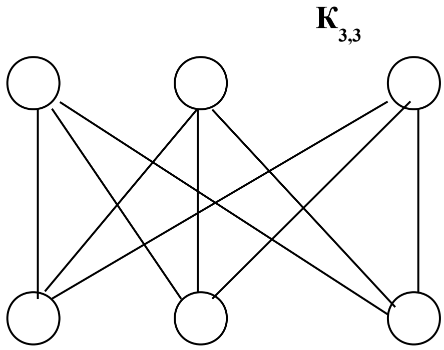
Доказательство. Без потери общности можно считать *G* плоским графом. Так как каждая грань ограничена по крайней мере тремя ребрами, то при подсчете числа ребер вокруг каждой из граней получим, что $\boldsymbol{3f \leq 2m}$ (множитель 2 появляется оттого, что каждое ребро ограничивает не больше двух граней). Это неравенство в сочетании с теоремой Эйлера и приводит к нужному результату.

3. Теорема Понтрягина-Куратовского

Задача 3 дома и 3 колодца

Утверждение 15 (Следствие). *Графы K_5 и $K_{3,3}$ не являются*





планарными.

Доказательство. Если K_5 планарен, то, применяя следствие 2, получим: $10\leq 9$, что, очевидно, невозможно. Чтобы показать, что $K_{3,3}$ непланарен, достаточно заметить, что каждая его грань ограничена по крайней мере четырьмя ребрами и любое ребро может принадлежать только 2-м разным граням => $\boldsymbol{4f\leq 2p}$ (т. е. $2\boldsymbol{f\leq 9}$). Но этого не может быть, поскольку теорема Эйлера утверждает ($\mathbf{S = n - p + f = 6 - 9 + f = 2}$), что число граней должно быть f=5.

Аналогичные рассуждения позволяют доказать следующую теорему, которая окажется полезной, когда мы перейдем к задачам о раскраске графов.

Утверждение 16. *В любом простом планарном графе существует вершина, степень которой не больше 5.*

Доказательство: метод от противного. Пусть в плоском графе все вершины С(к)=6. Тогда по следствию 2 получаем $\boldsymbol{p\leq 3n-6}$ и $\boldsymbol{\sum C=2p}$ => $\boldsymbol{\sum C\leq 6n-12}$ но тогда $\boldsymbol{\sum C\geq 6n}$ получаем противоречие => ч.т.д..

Теорема. (Понтрягина=Куратовского). *Граф планарен тогда и только тогда, когда он не содержит подграфов K_5 или $K_{3,3}$.*

Поскольку доказательство теоремы Куратовского довольно длинное и сложное проводить его мы не будем.

Теорема Фари

Для $\forall$ плоского графа $\exists$существует изоморфное изображение с помощью ребер – отрезков.

Теорема Тарри

Для $\forall$любого связного графа существует$\exists$ циклический маршрут через все ребра туда и обратно точно два раза.

Лекция №7

Раздел №3. Прикладные задачи теории графов

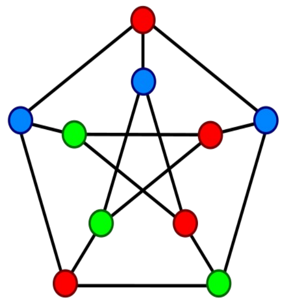
Содержание: Цветные графы. Свойства и применение цветных графов. Раскраска графов. Двудольные графы. Свойства и применение двудольных графов. Примеры.

1. Цветные графы

Прежде мы предполагали, что вершины, ребра, грани в графе одинаковы, однако, в некоторых случаях эти объекты могут быть разнотипными. Можно представить типы вершин, ребер или граней цветами. То есть раскрашенные в разные цвета вершины, ребра или грани и есть выделение типов. Принято говорить, что вводится раскраска графов, то есть цвет вершин, ребер, граней. При этом любой раскрашенный граф может быть представлен в виде раскраски вершин, если ребра или грани заменить некими вершинами. Такое преобразование исходного графа вершин в граф ребер или граф граней в принципе однозначен.

Цветные вершины

Определение. Правильной раскраской вершин называют такую раскраску, когда любые смежные вершины имеют разные цвета.



Пример правильной раскраски графов

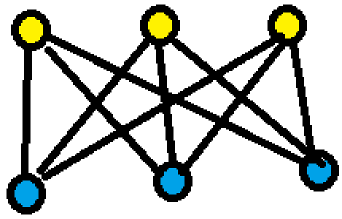
Определение. Хроматическое число χ графа – минимальное число красок для правильной раскраски вершин. Принято называть множество вершин одного цвета при правильной раскраске независимым множеством. Таким образом, алгоритм раскраски графа фактически является алгоритмом построения системы множеств независимых вершин. Очевидно, что такое построение набора независимых вершин не однозначно, но среди них обязательно есть минимальное число таких множеств, которые содержат все вершины графа. Это число и есть хроматическое число графа.

Определение хроматического числа графа — особая неоднозначная алгоритмическая задача, подобная поиску гамильтоновых циклов, где нет общих правил для всех возможных графов. Однако некоторые частные результаты были получены:

1.Хроматическое число полного графа равно числу вершин (доказательство очевидно).

2.Хроматическое число дерева равно 2, что следует из свойства дерева иметь единственный путь между любыми 2 вершинами. Такой путь можно раскрасить 2 цветами. Если на этом пути есть разветвления, то можно на разветвлении строить отрезок опять 2 цветами. В результате когда-то все вершины будут раскрашены.

3.Есть некоторые оценки χ графа, связывающие его с другими характеристиками графа. Например $\chi \leq k+1$, $\chi \leq (3+(9+8*(p-n))^{1/2})/2$, k- максимальная степень вершины, p — число ребер , n- число вершин графа. Тогда получаем : $(2\chi-3)^2 \leq 8*(p-n)+9$. Рассмотрим граф $K_{3,3}$, тогда p-n = 9-6=3, 8*3+9=24+9=33 => $2\chi-3 \leq 6$ => $\chi \leq 9/2$ или $\chi \leq 4$. Значит 4 цвета достаточно для правильной раскраски этого графа. На самом деле это двудольный граф, где в любом графе достаточно 2-х красок:



Так что эти оценки часто излишни.

4.Особую роль в задачах раскраски стали играть плоские графы. Это связано с задачей раскраски карты:

Дана политическая карта мира. Требуется раскрасить каждую страну в какой-либо цвет так, чтобы любые две граничащие между собой страны были раскрашены в разные цвета, использовав при этом минимально возможное число красок. Очевидно, что граф такой задачи плоский.

В 1890 году была доказана теорема Хивуда о 5 красках.

Теорема о 5 красках. Всякий планарный граф можно раскрасить 5 красками (задать цвет вершин так, чтобы смежные вершины не были одного цвета).

Доказательство. Т. к. граф плоский, то число ребер $p \leq 3n-6$. Степень хотя бы одной вершины должна быть 5 или меньше. Докажем методом математической индукции.

- Если число вершин – 3, то доказательство очевидно.
- Предположим, утверждение верно для $n=k$.
- Докажем утверждение для $n=k+1$. Добавим одну вершину. Тогда полученный граф тоже будет плоским и у него будет существовать вершина, степень которой ≤ 5. Удалим эту вершину и все смежные с ней ребра. Тогда оставшаяся часть графа тоже будет плоской и будет иметь k вершин. Следовательно, по индуктивному предположению, эту часть можно раскрасить 5 красками. Рассмотрим раскрас-

ку оставшейся вершины. Возможны варианты:

1. Степень вершины меньше 5, тогда для вершины всегда найдется свободный цвет.
2. Степень вершины = 5, но среди смежных вершин есть вершины одного цвета.
3. Число смежных вершин = 5, но цвета вершин не повторяются. В этом случае нам необходимо попытаться изменить цвета, чтобы получить правильную раскраску. Для перекраски нам необходимо рассмотреть простой цикл через данную вершину и 2 других из этих 5..Такой цикл обязательно будет, так как без данной вершины граф был связным. Перекраску будем производить внутри такого цикла . Для этого нам необходимо сделать так, чтобы 2 вершины, которые связаны с данной вершиной, имели бы одинаковый цвет. Если удалить одну из смежных вершин, а затем перекрасить граф так, чтобы другая смежная вершина, образующая с удаленной простую цепь также была окрашена в цвет удаленной вершины, то оставшийся граф будет иметь k вершин и являться плоским, следовательно, может быть раскрашен. При этом, у вершины со степенью 5 будет получен, не совпадающий с цветом удаленной вершины. То есть можно вернуть эту вершину в тот же самый граф с тем же цветом. Раскраска произошла.

Ч.т.д.

Кэлли предположил, что и 4 цвета достаточно для раскраски плоских графов (гипотеза о 4 красках). В 1976 году удалось с помощью ЭВМ проверить возможность раскраски 4 цветами, с тех пор эта гипотеза считается доказанной.

Сегодня считается, что общий алгоритм оптимальной раскраски произвольного графа так же сложен как задача коммивояжера (NP-алгоритм), поэтому используют приближенные алгоритмы (например жадный), которые не дают полной уверенности, что эта раскраска оптимальная.

Пример. Для раскраски плоских графов существует алгоритм последовательного раскрашивания:

1. Пусть задан граф G, рассмотрим в нем множество вершин, которые независимы, то есть, не соединены между собой. Среди таких множеств должно существовать множество вершин с наибольшим количеством элементов (множество С).
2. Покрасим вершины найденного множества в первый цвет.
3. Рассмотрим граф, полученный удалением вершин множества С и инцидентных к нему ребер из графа G. Полученный граф может быть связным или несвязным, но плоским.
4. Повторим процедуру 1-2 и используем второй цвет.

5. Повторим пункт 3 пока в графе не останется незакрашенных вершин.

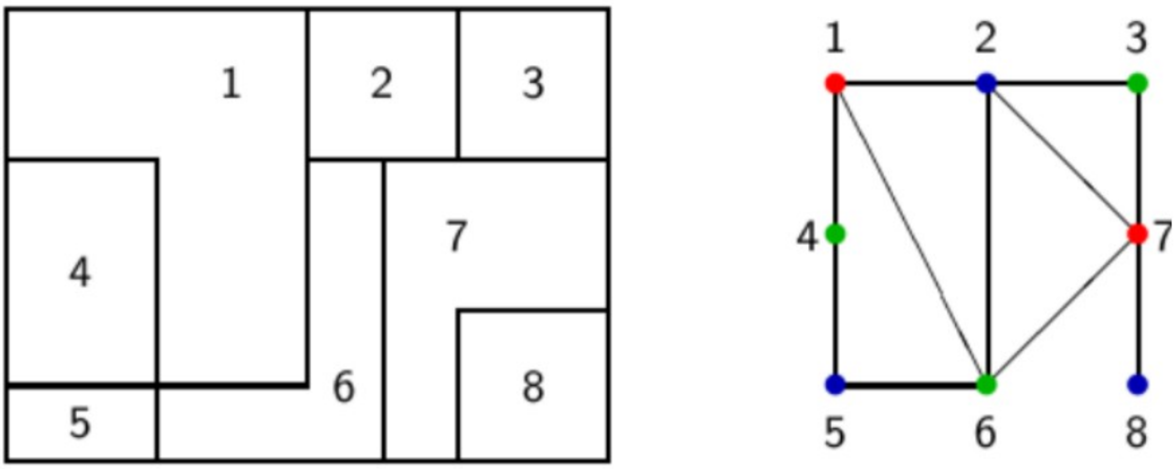
2. Применение цветных графов

При раскраске выделяют 3 задачи — раскраска вершин, раскраска ребер, раскраска граней. Рёберная раскраска графа подразумевает под собой назначение цветов ребрам так, что никакие два ребра одного цвета не принадлежат одной вершине. Наименьшее число цветов, необходимое для рѐберной раскраски графа G — это его хроматический индекс, или рёберное хроматическое число.

Тотальная (полная) раскраска — это один из видов раскраски вершин и рѐбер графа. Под ней подразумевают такое присвоение цветов, что ни соседние вершины, ни смежные ребра, ни вершины и ребра, которые их соединяют, не имеют одинакового цвета. Аналогично раскраска граней соответствует задаче раскраски карты, когда смежные участки карты не должны иметь одинаковый цвет. Полное хроматическое число (G) графа — это наименьшее число цветов, необходимое для любой полной раскраски.

Обычно все сводят к раскраске вершин, для этого нужно построить реберный граф (каждая вершина соответствует ребру) или двойственный граф (каждая вершина соответствует грани) и затем раскрашивают их вершины, что дает и раскраску ребер и граней.

Пример: На рисунке слева схематично изображена карта, а справа приведен соответствующий ей граф. Цифры означают номера стран.



Области применения задач раскраски графов

Планирование. Раскраска вершин моделирует многие проблемы планирования работ некоторого задания. В своей простейшей постановке заданный набор работ должен быть распределен по временным отрезкам, каждая такая работа занимает один отрезок. Они могут быть выполнены в любом порядке, но две работы могут конфликтовать в том смысле, что не могут быть выполнены одновременно, так как, например, используют общие ресурсы. Соответствующий граф содержит вершину для каждой из работ и ребро для каждой конфликтующий пары. Хроматическое число построенного графа — это минимальное время выполнения всех работ без конфликтов. Детали проблемы планирования определяют структуру графа. Например, когда идет распределение самолетов по рейсам, результирующий граф конфликтов является интервальным графом, так что проблема раскраски может быть решена эффективно. При распределении радиочастот получается граф единичных кругов конфликтов, и для такой задачи существует 3-аппроксимационный алгоритм.

Распределение регистров. Компилятор — это компьютерная программа, которая переводит один компьютерный язык в другой. Для улучшения времени выполнения результирующего кода одной из техник компиляторной оптимизации является распределение регистров, в которой наиболее часто используемые переменные компилируемой программы хранятся в быстродействующих регистрах процессора. В идеальном случае переменные хранятся в регистрах так, что они все находятся в регистрах во время их использования. Стандартный подход к этой задаче состоит в сведении её к задаче раскраски графов. Компилятор строит граф взаимодействия, где вершины соответствуют регистрам, а грань соединяет две из них, если они нужны в один и тот же момент времени. Если этот граф k-хроматический, то переменные могут храниться в k-регистрах.

Цифровые водяные знаки. Технология цифровых водяных знаков (англ. digital watermarking) позволяет вместе с данными (будь то медиафайлы, исполняемые файлы и прочие) передать некое скрытое сообщение («водяной знак», Watermark). Такое скрытое сообщение может быть применено в защите авторских прав для идентификации владельца данных. Это важно, например, для установления источника их распространения нелегальным образом. Или же для подтверждения прав на данные, например — программное обеспечение систем на кристалле (system-on-chip). Сообщение можно закодировать в том числе и в способе распределения регистров (например QP-алгоритм). Оказывается, возможно закодировать послание в программном продукте с помощью раскраски графа G, то есть распределения регистров. Извлечь это сообщение можно путѐм сравнения распределения регистров с исходной раскраской

Задача составления расписания. Нужно прочесть несколько занятий с несколькими группами студентов. Некоторые из занятий не могут проводиться одновременно (например, потому, что их проводит один и тот же преподаватель, или их надо проводить в одной и той же группе студентов, или они должны проходить в одном и той же аудитории. Требуется составить расписание так, чтобы проведение всех занятий заняло минимально возможное время (то есть без «дырок» в расписании и за минимальное число дней).

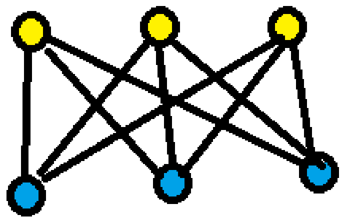
Задача распределения оборудования. Это задача связана с распределением оборудования при выполнении какой-то работы. Имеется некоторое количество работ и механизмов для их осуществления. Для выполнения каждой работы требуется одно и то же время. При этом никакой из механизмов не может быть занят одновременно более чем в одной работе. Нужно распределить механизмы так, чтобы общее время выполнения работ было минимально возможным.

3. Двудольные графы

Двудольные графы и паросочетания

О. Граф называется *двудольным*, если существует такое разбиение множества его вершин на два непересекающихся подмножества (на две доли), что концы каждого ребра принадлежат разным подмножествам (разным долям).

Пример (уже известный граф):



Произвольное подмножество попарно несмежных рёбер графа (взятых вместе со своими вершинами) называется *паросочетанием* (или *независимым множеством рёбер).*

О. Паросочетание – множество ребер графа не имеющих общих вершин.

Цепь (чередущая) – объединение поочередносвязанных ребер паросочетания и нет.

Паросочетание M наибольшее тогда и только тогда когда нет для M увеличивающих цепей.

Совершенное паросочетание – множество вершин V_1 и V_2 такое, что все вершины V1 входят в это паросочетание.

Совершенное паросочетание всегда является максимальным (то есть наибольшее для данного графа).

Теорема Холла

Определение. Пусть $A=\{a_1,...,a_n\}$ конечное непустое множество, а $S = (S_1,...,S_k)$ — k-членное семейство его подмножеств (необязательно непересекающихся и необязательно различных). Всякое подмножество $T=\{a_{i_1},....,a_{i_k}\}$элементов из A называется ***трансверсалью*** (или *системой различных, представителей)* семейства S, если $a_{i_j} \in S_j, 1 \le j \le k, i_j \neq i_{j'}$ при $j \neq j'$.

Теорема (Холла). Пусть A — непустое конечное множество, $S = (S_1,$

$S_2,..., S_k)$ — семейство его подмножеств. Для существования трансверсали семейства S необходимо и достаточно, чтобы для каждого j = 1,..., k объединение $S_{i_1} \cup S_{i_2} \cup ... \cup S_{i_j}$ любых j подмножеств S_i содержало не менее j различных элементов.

Если теперь сравнить теорему (критерий) о существовании паросочетания, покрывающего долю вершин в двудольном графе, с теоремой Холла, то нетрудно заметить, что первая является переформулировкой второй на языке графов (элементы множества A образуют одну долю W вершин графа, подмножества семейства S образуют другую долю V — покрываемую паросочетанием).

Теорема Холла имеет множество различные применений и интерпретаций.

4. Применение двудольных графов

С построением паросочетаний в двудольном графе связано много самых разных задач. Типичные примеры — задача о назначении на должности, задача о свадьбах, задача о продажах и т.д. В уже известной нам задаче «3 дома, 3 колодца» тоже граф с 2 долями (дома и колодцы).

Очень известна в приложениях «Задача о максимальном паросочетании» – поиск максимального числа ребер в паросочетании.

В теории искусственного интеллекта очень часто используются сети Петри — двудольные графы состояний с заданными вероятностями переходов между состояниями.

В теории кодирования некоторые важные коды строятся на основе деревьев. При анализе или коррекции такого дерева могут быть построены двудольные графы коррекции ошибок.

Лекция №8-9-10

Раздел №3. Прикладные задачи теории графов

Содержание: Задача Прима-Краскала. Алгоритмы Прима и Краскала. Задача Дейкстры. Алгоритмы Дейкстры и Уоршелла. Транспортная сеть. Потоки в транспортной сети. Задача о максимальном потоке. Теорема Форда-Фалкресона. Понятие о задаче сетевого планирования.

1. Задача Прима-Краскала

О. Задача Прима-Краскала связана с поиском остовного дерева графа минимальной длины. Граф должен иметь длины ребер, быть связным. Длиной графа будем считать сумму длин его ребер.

Данная задача относится к задачам решаемым «жадным» алгоритмом (задача на матроиде). Решить «жадным» алгоритмом означает, что для поиска максимума или минимума мы должны на каждом шаге выбирать наилучшее (максимальный или минимальный) вариант возможного хода. Например для поиска остовного дерева можно использовать алгоритм:

Шаг 1. Изображаем граф без ребер, только вершины.

Шаг 2. Находим самое короткое ребро и добавляем его к изображению.

Шаг 3. Среди оставшихся ребер находим все ребра, у которых одна вершина не связана с уже построенным изображением, а вторая связана. Такое ребро не создает цикла и может быть использовано для построения остовного дерева. В этом множестве ребер находим самое короткое ребро и добавляем его к изображению. Таким образом, каждый раз выполняя шаг №3 мы строим дерево, присоединяя к уже связанной части новую вершину.

Шаг 4. Повторяем шаг №3 до тех пор, пока не свяжем все вершины. В результате получим остовное дерево исходного графа. Поскольку исходный граф был связан, то для любой вершины найдутся ребра связывающие ее с другой частью этого дерева.

Этот алгоритм построения остовного дерева получил название алгоритма Прима.

Теорема Прима. Остовное дерево связного, взвешенного графа с заданными длинами ребер построенное по «жадному» алгоритму Прима будет иметь минимальную длину.

Док-во: Используем метод мат.индукции по числу ребер графа. Для графа с n=2 или 3 доказательство очевидно.

Пусть теперь теорема Прима доказана для любого связанного графа из n ребер. Докажем тогда, что она будет справедлива и для любого графа из n+1 ребра.

Во первых, докажем, что в таком графе всегда можно удалить какую-либо вершину вместе со смежными ребрами так, чтобы новый граф оставался связным. В графе либо есть циклические ребра, либо это дерево. Если граф является деревом, то у него есть висячая вершина. Которую и можно удалить. Если граф не дерево и у него нет висячих вершин, то в нем есть циклы. Будем находить циклы и удалять из каждого 1 циклическое ребро, связность при этом не нарушается и число вершин остается прежним. Очевидно, что на каком-то шаге удаления получим дерево, а значит и висячую вершину. Рассмотрим эту висячую вершину и ребра которые удалили перед этим. Так ранее эта вершина не была висячей, то среди удаленных ребер есть ребра инцендентные этой вершины. Если удалить эту вершину и ребра инцендентные этой вершине, то граф останется связным (так как все ребра кроме одного уже удалялись не приводя к разрыву связи).

Итак после удаления вершины и инцендентных этой вершине ребер мы получили связный граф с n вершинами, в котором остовное дерево минимальной длины строится алгоритмом Прима. Пусть теперь мы строим остовное дерево для исходного графа алгоритмом Прима и сравниваем его с истинным остовным деревом минимальной длины этого графа. Назовем удаленную ранее вершину буквой М. М входит в истинное дерево либо как висячая вершина, либо нет. Если вершина М не висячая, то можно всегда выбрать любую висячую вершину К и поменять ее с М местами. То есть удалить ее из исходного графа с инцендентными этой вершине ребрами. Таким образом, истинное дерево после удаления вершины К и ребра с ней связанного будет остовным деревом графа n вершин. Причем это должно быть минимальное дерево, так как вершина К добавляет в сумму длину только 1 ребра и не влияет на выбор остальных ребер дерева. По условию теоремы остовное дерево графа с n ребрами такой длины может быть построено алгоритмом Прима. Если же мы применим алгоритм Прима к вершине К, то оно добавится к этому дереву с минимальным ребром, которое свяжет ее с основным деревом. Очевидно, что общая сумма не будет превышать длину истинного дерева. То есть построенное алгоритмом Прима остовное дерево граф с n+1 ребром будет иметь длину не более длины истинного минимального дерева. А это возможно только при совпадении длин.

По методу мат.индукции для любого числа вершин N теорема верна. Ч.Т.Д.

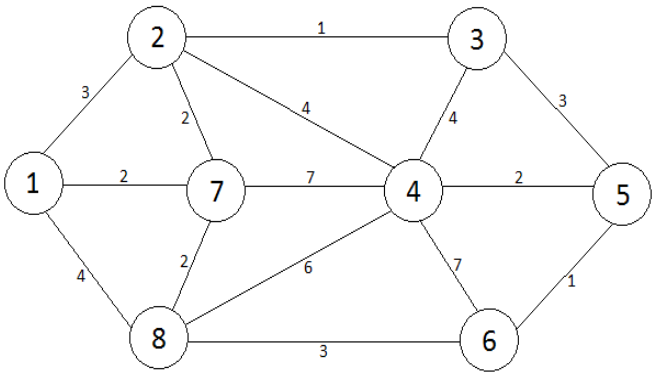
Замечание. В теореме не утверждается что остовное дерево минимальной длины единственное. Для многих реальных графов помимо остовного дерева Прима можно построить другие остовные деревья такой же длины.

2. Алгоритмы Прима и Краскала

По алгоритму Прима искомое остовное дерево получается путем последовательного подсоединения к вершинам графа его ребер. На первом шаге все ребра удаляются. Затем ребра сортируются в порядке возрастания длины. Первое добавленное ребро – кратчайшее, далее ребра выбираются по особой процедуре:

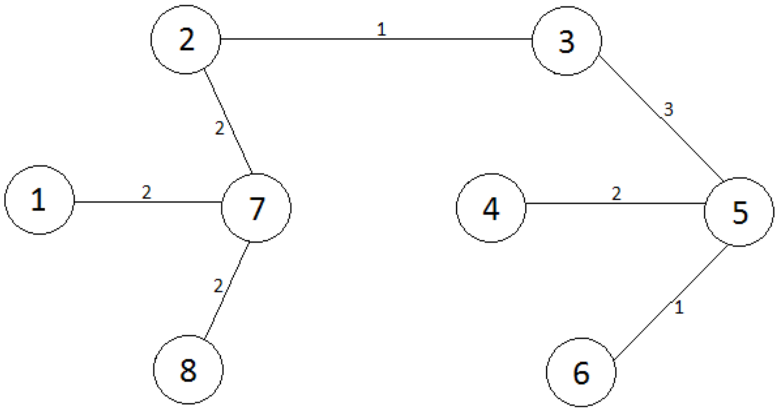
- Выделяется множество вершин графа которые уже связаны с установленными ребрами (множество А). Множество В – все остальные вершины.
- Выделяем множество ребер, которые соединяют вершины множеств А и В, но не соединяют вершины одного множества.
- Из выделенного множества ребер находим кратчайшее, его добавляем в граф.

Алгоритм повторяют до тех пор, пока множество В не станет пустым.
Рассмотрим алгоритм на примере графа:



Перепишем ребра в порядке возрастания. Строим граф согласно алгоритму.

23	1+
56	1+
17	2+
27	2+
78	2+
45	2+
12	3-
35	3+
68	3
24	4
18	4
34	4
48	6
46	7
47	7



Алгоритм Краскала

Алгоритм Краскала более сложен для реализации. Его действие противоположно алгоритму Прима. Берем исходный граф со всеми существующими ребрами и затем последовательно удаляем циклические ребра. Проблема здесь в том, что нельзя просто удалить длиннейшее ребро, надо сначала убедиться, что оно циклическое. Поэтому нужно в списке ребер проверять цикличность и первое циклическое ребро в списке ребер по убыванию удалять.

Шаг 1. Изображаем связный граф.

Шаг 2. Находим самое длинное циклическое ребро и удаляем его из графа. Граф остается связным.

Шаг 4. Повторяем шаг №2 до тех пор, пока не останется n-1 ребро, где n -число вершин. В результате получим остовное дерево исходного графа.

Этот алгоритм построения остовного дерева получил название алгоритма Краскала.

Теорема Краскала. Остовное дерево связного, взвешенного графа с заданными длинами ребер построенное по «жадному» алгоритму

Краскала будет иметь минимальную длину.
(без доказательства)
Рассмотрим алгоритм на примере того же графа. Для алгоритма Краскала процесс обратный, но дерево получится в примере то же.

Замечание. Так ка остовное дерево минимальной длины может быть не единственным, то для реальных графов остовного дерева Прима может отличаться от остовного дерева Краскала.

3. Задача Дейкстры

Задача Дейкстры связана с поиском кратчайших путей между вершинами графа при наличии длин ребер. Практически нужно найти кратчайшие расстояния и соответствующие им пути от некой начальной вершины А до всех остальных вершин графа.

Рассмотрим алгоритм:

Шаг №1. Строим исходный связный граф и выбираем вершину А. Помечаем А цифрой 0.

Шаг №2. Начинаем разметку вершин графа с вершин прямо связанных общим ребром с А. Ставим у таких вершин метку равную длине ребра от А. Метка определяет длину пути от А до этой вершины.

Шаг №3. Оптимизируем метки вершин графа. Для этого последовательно проверяем вершины с прямо связанными с ними общим ребром вершинами. Проверку ведем по «жадному» алгоритму. Выбираем не проверенную вершину с минимальной меткой. Проверяем сумму этой метки+длина общего ребра, если эта сумма меньше метки вершины с которой общее ребро, то ее метка изменяется на новую. Ставим метку так же у вершин где раньше метки не было. Таким образом, метки определяют возможную длину пути не только прямо из А, но и через другие вершины. Чтобы не потерять последовательность этого пути желательно у метки ставит букву вершины, через которую эта метка найдена. При этом, метки и буквы вершин при выполнении алгоритма могут многократно меняться.

Этот алгоритм получил название алгоритма Дейкстры.

По размеченному графу алгоритма Дейкстры можно построить остовное дерево следующим образом:

Шаг 1. Изображаем граф без ребер, только вершины.

Шаг 2. По разметке графа находим вершины самый короткий путь к которым составляет прямое ребро от А и добавляем эти ребра к изображению.

Шаг 3. По разметке графа находим вершины самый короткий путь к которым составляет прямое ребро от уже связанным вершинам и добавляем эти ребра к изображению.

Шаг №3 повторяется пока все вершины не будут связаны. Полученное дерево называют деревом Дейкстры и оно показывает кратчайшие пути от

А до других вершин (так как в дереве между А и любой другой вершиной есть только 1 путь, то он и есть кратчайший, а сумма длин его ребер — минимальная длина пути).

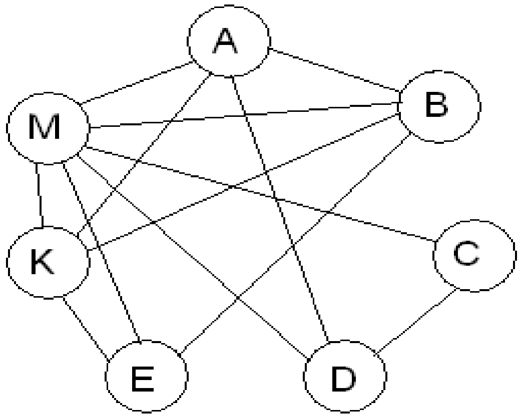
Теорема Дейкстры. Построенное по «жадному» алгоритму Дейкстры остовное дерево связного, взвешенного графа с заданными длинами ребер будет содержать кратчайшие пути от исходной вершины ко всем остальным вершинам графа.

Док-во: Метод математической индукции по номеру вершин в дереве. Запишем на основе дерева Дейкстры массив, содержащий кратчайшие пути к вершинам в виде набора составляющих их взвешенных ребер, который отсортирован в порядке увеличения длины пути. Каждый элемент массива соответствует вершине дерева. Первый элемент — начальная вершина А, где длина пути равна 0. Пусть n – это номер вершины в этом массиве, тогда для А(n =0). Далее вершины имеют n =1, 2 и т.д. Докажем что для n =1 вершина дерева Дейкстры имеет кратчайший путь. Это вершина, которая прямо связана с А и имеет кратчайшее ребро. Очевидно, что по алгоритму Дейкстры она получит именно этот кратчайший путь. Пусть теперь на месте n =К расположен кратчайший путь, докажем что и при К+1 тогда в массиве будет кратчайший путь. На этом месте будет расположена вершина с той же длиной пути или большей. Если длина та же, то и путь кратчайший. Пусть длина увеличилась, тогда метод от противного. Пусть есть другая истинная вершина в таком массиве. 2 варианта — вершина эта же и вершина другая. Если вершина эта же, то она имеет путь короче пути в дереве Дейкстры. Например последнее ребро в этом пути идет от вершины В. Тогда вершина В должна быть в массиве раньше чем К+1, но тогда ее длина пути в дереве Дейкстры минимальна. Следовательно, длина пути от А до В + от В ло текущей вершины проверяется алгоритмом Дейкстры => этот вариант невозможен. Второй вариант — вершина другая, но тогда она тоже должна быть в массиве => опять сравниваем пути с массивом и получаем противоречие. Итак при К+1 в массиве Дейкстры будет располагаться кратчайший путь => по методу матиндукции Ч.Т.Д.

4. Алгоритмы Дейкстры и Уоршелла

Рассмотрим алгоритм Дейкстры на примере

Задан граф:



Заданы длины ребер графа. AB=7 AD=12 AM=5 BE=11 BK=9 BM=7 CM=6 CD=10 EM=5 EK=12 DM=5 KM=6 АК=6. Найти кратчайшие расстояния от вершины А до всех остальных вершин графа. Всего 7 вершины:A, B, C, D, E, K, M. Начальная вершина А.

Шаг 1. Строим метки ближайших вершин (прямой путь) А=0, В=7, М=5, К=6, D=12. Выбираем минимальную метку (кроме А) это М. Строим первое ребро дерева Дейкстры — это ребро АМ.

Шаг 2. Строим метки с учетом ближайших вершин от М (проверяем вершины где меток нет или они больше метки М). Для В=7<5(М)+7(ВМ), К=6<5(М)+6(КМ) — эти метки не меняются. Для D=12>5(M)+5(DM) — метку меняем на 10. Метки новых вершин будут С=5(М)+6(СМ)=11, Е=5(М)+5(ЕМ)=10.

Итоговые метки А=0, М=5, К=6, В=7, D=10, E=10, C=11. Выбираем минимальную метку (кроме А и М) это К. Добавляем новое ребро дерева Дейкстры — это ребро АК.

Шаг 3. Аналогично шагу 2 строим метки с учетом ближайших вершин от К (проверяем вершины где метки больше метки К).

Итоговые метки А=0, М=5, К=6, В=7, D=10, E=10, C=11. Выбираем минимальную метку (кроме А,М и К) это В. Добавляем новое ребро дерева Дейкстры — это ребро АВ.

Далее метки уже меняться не будут. Добавляются в дерево ребра MD , ME, MC.

Итоговые пути А-М=5, А-К=6, А-В=7, А-М- D=10, А-М- E=10, А-М-C=11,

Алгоритм Уоршелла – усовершенствование метода Дейкстры для решения задачи поиска всех расстояний между всеми вершинами. Его можно представить в виде программы Паскаля:

```
For k:=1 to N do
For i:=1 to N do
For j:=1 to N do
d[i,j]:= min(d[i,j], d[i,k]+d[k,j]);
```

Здесь d[i,j] – матрица расстояний, min – функция вычисляющая минимальное из 2 значений.

Алгоритм Уоршелла (Уорхолла, Воорхолла) основан на теореме о построении матрицы расстояний на основе проверки треугольников вершин А-В-С-А, В таком треугольнике выбирается минимальное из АВ и АС+ВС для оценки пути из А до В.

5. Задача Форда-Фалкерсона

Сетью называется связный граф с выделенной парой вершин – входом и выходом. Если есть несколько входов или выходов, то создают виртуальный вход или выход, соединенный со всеми входами или выходами. Транспортной сетью называется сеть (чаще всего с дугами, как ориентированный граф) с заданной на дугах специальной функцией – пропускной способностью (которая должна быть положительной).

Вершина начала транспортной сети, из которой дуги только выходят. Вершина выхода транспортной сети, в которую дуги только входят. На множестве дуг *b* задана целочисленная функция *c(b)* > 0, где *c(b)* – пропускная способность дуги *b*.

Определение. Потоком по транспортной сети называется положительно определенная функция φ(b)≥0 , заданная на множестве дуг *b* и обладающая следующими свойствами:

А) Для любой дуги *c(b)* ≥ φ(b). Б) Сумма входных потоков в любой вершина сети равна сумме выходных потоков.

Свойство Б утверждает, что поток, входящий в вершину, равен выходящему потоку (поток в вершинах не скапливается). Это свойство называют правилом (законом) Кирхгофа.

Дуга называется насыщенной если для нее c= φ.

Набор дуг, удаление которых ведет к разрыву связи между входом и выходом (связь есть, если есть путь между вершинами) называют сечением (разрывом) сети.

Простым сечением называют сечение в котором нельзя удалить ни одну дугу без потери свойства сечения (нет лишних дуг).

Насыщенным сечением является сечение в котором все дуги насыщены.

Пропускной способностью сечения называют сумму пропускных способностей ее дуг.

Задача Форда- Фалкерсона (ФФ) – поиск максимально возможного потока в транспортной сети.

Теорема Форда- Фалкерсона – максимальный поток в сети равен минимальной пропускной способности сечений этой сети.

(без доказательства)

Методы решения задачи Форда-Фалкерсона

Для решения задачи нужно найти набор дуг (сечение), которое обладает следующими свойствами:

- Все дуги набора(сечения) насыщены.
- Удаление дуг набора(сечения) приводит к разрыву связей между входом и выходом.

- Для разрыва связи нужны все дуги набора(сечения). Если удалить любую дугу, то сечение не будет удовлетворять 2-му свойству.

Поиск решения имеет множество различных алгоритмов. Рассмотрим 2 из них:

А) Метод обратного планирования Беллмана. Суть метода определяется алгоритмом:

- Выбираем выходную вершину как слой номер Т.
- Рассматриваем все дуги, входящие в слой Т. Насыщаем их.
- Рассматриваем все вершины, откуда начинаются дуги, входящие в слой Т. Определяем их как слой Т-1.
- Рассматриваем все дуги, входящие в слой Т-1. Определяем их потоки, так чтобы для вершин слоя Т-1 выполнялось правило Киргофа. Если это не удается, то нужно уменьшить выходные потоки в вершины слоя Т.
- Алгоритм повторяется до тех пор пока потоки всех дуг не будут определены.
- Результат распределения должен быть максимальным потоком. Убеждаемся в этом находя насыщенное сечение сети. Поток в слое Т будет тогда равен максимальному потоку.

Б) Метод последовательного перехода. Суть метода определяется алгоритмом:

- 1) Находим опорный поток. Например выбираем поток равный 0 везде.
- 2) Находим путь П между входом и выходом, в котором все дуги не насыщенны.
- 3) Находим дугу В на пути П для которой разница с-φ минимальна.
- 4) Увеличиваем поток по дуге П так, чтобы дуга В была насыщенна.
- 5) Помечаем насыщенную дугу В (можно даже для наглядности удалять ее из сети).
- 6) Добавляем В к списку насыщенных дуг и проверяем не является ли полученный список сечением. Если список сечение, то проверяем – нельзя ли удалить из него дуги и получить простое сечение. Если список еще не сечение, то повторяем п.3-6.
- 7) Вычисляем пропускную способность полученного простого сечения. По теореме ФФ эта величина равна максимальному потоку.

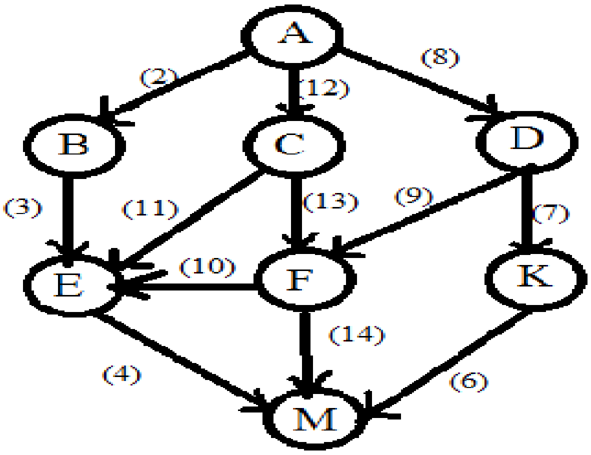
Пример : Задача Форда-Фалкерсона с построением графа сети.

Пропускные способности ребер сети:

AB=2, AD=8, AC=12, BE=3, CE=11, CF=13, DF=9, DK=7, FE=10, EM=4, FM=14, KM=6

Решение:

Построим изображение графа по данным задания (см. рисунок 35).
Проверив степени входа и выхода вершин, определяем вход и выход. Вход А, выход М.



Шаг № 1. Начальный поток P=0
Ищем путь от входа к выходу. Путь: A->B->E->M
Находим в этом пути самую минимальную пропускную способность(с учетом потока). Min: AB=2 ==>увеличиваем P на 2 P=P+2 ==> Насыщаем AB
Шаг № 2. Поток P=2
Ищем путь, который не проходит через насыщенные ребра. Путь: A->D->K->M
min: KM=6 ==>Увеличиваем P на 6 P=P+6 ==>Насыщаем KM
Шаг № 3. Поток P=8 Путь A->C->F->M min: AC=12 P=P+12 ==> Насыщаем AC
Шаг № 1. Поток P=20. Путь: A->D->F->M min: AD=2 P=P+2 ==> Насыщаем AD, FM. Найдено насыщенное сечение
S=AC+AD+AB=2+12+8=22 => по теореме Форда-Фалкерсона максимальный поток 22.
Ответ: P=22.

6.Задача сетевого планирования
Структура, представление и характеристики моделей СПУ
Назначение и области применения сетевого планирования и управления

Планирование сложных процессов потребовало создания специальных методов сетевого планирования и управления (СПУ). В основе методов СПУ лежит применение сетевых графиков. Первые системы СПУ, использующие сетевые графики, появились в конце 1950-х годов в США и ныне известны по аббревиатурам CPM (Critical Path Method – метод критического пути) и PERT (Program Evaluation and Review Technique

– метод анализа и оценки программ). СПУ основано на моделировании процесса с помощью сетевого графика и представляет собой совокупность расчетных, организационных и контрольных мероприятий по планированию и управлению комплексом работ.

Комплекс работ (***комплекс операций***, или ***проект***) – любая достаточно сложная задача, решение которой требует выполнения большого количества разнообразных работ (например, строительство здания, корабля, самолета или любого другого сложного объекта).

Система СПУ позволяет:

- формировать календарный план реализации некоторого комплекса работ;
- – выявлять и мобилизовывать резервы времени, трудовые, материальные и денежные ресурсы;
- – осуществлять управление комплексом работ по принципу «ведущего звена» с прогнозированием и предупреждением возможных срывов в ходе работ;
- – повышать эффективность управления в целом при четком распределении ответственности между руководителями разных уровней и исполнителями работ.
- Для составления плана по реализации комплекса работ необходимо составить его математическую модель, в качестве которой в данном случае выступает *сетевая модель*.

Основные понятия сетевой модели

Сетевая модель – это план выполнения некоторого комплекса работ, заданный в специфической форме сети (дугам поставлены в соответствие интервалы времени), графическое изображение которой называют *сетевым графиком*. Главными элементами сетевой модели являются *события* и *работы*. Термин ***работа*** используется в СПУ в широком смысле. Во-первых, это *действительная работа* – протяженный во времени процесс, требующий затрат ресурсов (например, сборка изделия, испытание прибора и т.п.). Каждая действительная работа должна быть конкретной, четко описанной и иметь ответственного исполнителя. Во-вторых, это *ожидание* – протяженный во времени процесс, не требующий затрат труда (например, процесс сушки после покраски, отвердения бетона и т.п.).

В-третьих, это *зависимость*, или *фиктивная работа*, – логическая связь между двумя или несколькими работами (событиями), не требующими затрат труда, материальных ресурсов или времени. Она указывает, что возможность выполнения одной работы непосредственно зависит от результатов другой. Продолжительность фиктивной работы принимается равной нулю.

Событие – это момент завершения какого-либо процесса, отражающий отдельный этап выполнения проекта. Событие может быть частным результатом отдельной работы или суммарным результатом выполнения нескольких работ. Событие может свершиться только тогда, когда закончатся все работы, ему предшествующие. Последующие работы могут начаться только тогда, когда событие свершится. Таким образом, события имеют двойственный характер: для всех непосредственно ему предшествующих работ оно является *конечным*, а для всех непосредственно следующих за ним – *начальным*. При этом предполагается, что *событие не имеет продолжительности и свершается как бы мгновенно*. Поэтому каждое событие, включаемое в сетевую модель, должно быть полно, точно и всесторонне определено, его формулировка должна включать в себя результат всех непосредственно предшествующих ему работ. Среди событий сетевой модели выделяют *исходное* и *завершающее* события. Исходное событие не имеет предшествующих работ и событий, а завершающее – последующих работ и событий. Сетевая модель графически представляется в виде ориентированного графа, вершины которого являются событиями, а дуги – работами.

Необходимо отметить, что принцип построения сетевого графика может отличаться от описанного выше и основанного на сети вида «события–работы». Сетевой график может быть представлен сетью вида «работы–связи», вершины которой обозначают работы, а дуги – связи между работами, определяющими порядок их выполнения.

Следует отметить, что сетевой график «работы–связи» в отличие от графика «события–работы» обладает известными преимуществами: не содержит фиктивных работ, имеет более простую технику построения и перестройки, использует хорошо знакомое исполнителям понятие «работа», не включая менее очевидного понятия «событие». Вместе с тем сети без использования событий оказываются более громоздкими, так как событий обычно значительно меньше, чем работ (*показатель сложности сети*, равный отношению числа работ к числу событий, как правило, существенно больше 1). Поэтому эти сети менее эффективны с точки зрения управления комплексом работ и не получили достаточно широкого распространения.

Особенности решения оптимизационных задач в моделях СПУ

Порядок и правила построения сетевых графиков

Сетевые графики составляются на начальном этапе планирования путем разбиения планируемого процесса на отдельные работы, составления перечня работ и событий, установления их логической связи и последовательности выполнения, закрепления ответственных исполнителей за отдельными работами, оценки (совместно с ответственными исполнителями) длительности каждой работы. После построения первоначального сетевого графика выполняется его упорядочивание, рассчитываются параметры событий и работ, определяются резервы времени и *критический путь*, проводится анализ и оптимизация. После этого сетевой график может быть построен заново с использованием рассчитанных параметров для событий и работ. При построении сетевых графиков необходимо соблюдать следующие правила:

1. В сетевой модели не должно быть «тупиковых» событий, т.е. событий, из которых не выходит ни одна работа, за исключением завершающего события.
2. В сетевом графике не должно быть «хвостовых» событий, т.е. событий, которым не предшествует хотя бы одна работа, за исключением исходного. Эти ошибки так же называют тупиками.
3. В сети не должно быть контуров и петель, т.е. путей, соединяющих некоторые события с ними же самими.
4. Любые два события должны быть непосредственно связаны не более чем одной работой (стрелкой). Чтобы выполнить это требование, в некоторых случаях приходится вводить *фиктивное событие* и *фиктивную работу*, изображаемую на графике пунктирной линией.
5. В сети рекомендуется иметь одно исходное и одно завершающее событие.

Упорядочение сетевого графика

Упорядочение сетевого графика заключается в таком расположении событий и работ, при котором для любой работы предшествующее ей событие расположено левее и имеет меньший номер по сравнению с завершающим эту работу событием. Другими словами, в упорядоченном сетевом графике все работы-стрелки направлены слева направо: от событий с меньшими номерами к событиям с большими номерами.

Одним из важнейших понятий сетевого графика является понятие пути.

Путь – любая последовательность работ, в которой конечное событие каждой работы совпадает с начальным событием следующей работы. Среди различных путей сетевого графика наибольший интерес представляет *полный путь* – любой путь, начало которого совпадает с исходным событием сети, а конец – с завершающим. Наиболее продолжительный путь в сетевом графике называется *критическим*. Критическими называются также работы и события, расположенные на этом пути.

Критический путь имеет особое значение в системе СПУ, так как работы этого пути определяют общую продолжительность работы над проектом. Для сокращения продолжительности проекта необходимо в первую очередь сокращать продолжительность работ, лежащих на критическом пути.

Следует отметить, что традиционный вид сетевого графика, дающий четкое представление о порядке следования работ, не предусматривает использования масштаба времени, что не позволяет сразу определить те работы, которые должны выполняться в каждый данный момент времени. Поэтому сетевой график рекомендуется дополнять линейной диаграммой (называемой также графиком привязки) проекта.

Временные параметры сетевых графиков

Основные временные параметры сетевого графика:

- **Ранний срок свершения события**
- **Поздний срок свершения события**
- **Резерв времени события**
- **Продолжительность работы**
- **Ранний срок начала работы**
- **Ранний срок окончания работы**
- **Поздний срок начала работы**
- **Поздний срок окончания работы**
- **Полный резерв времени работы**
- **Частный резерв времени работы**
- **Свободный резерв времени работы**
- **Независимый резерв времени работы**

Рассмотрим содержание и способы расчета указанных временных параметров сетевого графика. Начнем с параметров событий. Событие не может наступить прежде, чем завершатся все предшествующие работы. Поэтому *ранний* (*или ожидаемый*) *срок свершения события определяется продолжительностью максимального пути, предшествующего этому событию*: Задержка свершения события по отношению к своему раннему сроку не отразится на сроке свершения завершающего события (а значит, и на сроке выполнения комплекса работ) до тех пор, пока сумма срока свершения этого события и продолжительности (длины) максимального из последующих за ним путей не превысит длины критического пути.

Резерв времени события определяется как разность между поздним и ранним сроками его свершения: Резерв времени события показывает, на какой допустимый период времени можно задержать наступление этого события, не вызывая при этом увеличения срока выполнения комплекса работ. Критические события резервов времени не имеют, поэтому любая задержка в свершении события, лежащего на критическом пути, вызовет такую же задержку в свершении завершающего события. Следовательно, чтобы определить длину и топологию критического пути, вовсе не обязательно перебирать все полные пути сетевого графика и определять их длины. *Достаточно определить ранний срок наступления завершающего события сети*, *тем самым*, *определив длину критического пути*, *а затем*, *выявив события с нулевыми резервами времени*, *определить его* *топологию*.

Среди резервов времени работ выделяют четыре разновидности.

Полный резерв времени работы показывает, на сколько можно увеличить время выполнения данной работы при условии, что срок выполнения комплекса работ не изменится. Полный резерв времени работы равен резерву максимального из путей, проходящего через данную работу. Этим резервом можно располагать при выполнении данной работы, если ее начальное событие совершится в самый ранний срок, и можно допустить свершение конечного события в его самый поздний срок. Важным свойством полного резерва времени работы является то, что он принадлежит не только этой работе, но и всем полным путям, проходящим через нее. При использовании полного резерва работы только для одной работы резервы времени остальных работ, лежащих на максимальном пути, проходящем через нее, будут полностью исчерпаны. Резервы времени работ, лежащих на других (не максимальных по длительности) путях, проходящих через эту работу, сократятся соответственно на величину использованного резерва.

Остальные резервы времени работы являются частями ее полного резерва.

Частный резерв времени работы есть часть полного резерва времени, на которую можно увеличить продолжительность работы, не изменив при этом позднего срока ее начального события. Этим резервом можно располагать при выполнении данной работы в предположении, что ее начальное и конечное события свершаются в самые поздние сроки

Свободный резерв времени *работы представляет часть полного резерва времени, на которую можно увеличить продолжительность работы, не изменив при этом раннего срока ее конечного события.* Этим резервом можно располагать при выполнении данной работы в предположении, что ее начальное и конечное события свершатся в свои самые ранние сроки.

Независимый резерв времени работы – часть полного резерва времени, получаемая для случая, когда все предшествующие работы заканчиваются в поздние сроки, а все последующие работы начинаются в ранние сроки. Использование независимого резерва времени не влияет на величину резервов времени других работ. Независимые резервы стремятся использовать тогда, когда окончание предыдущей работы произошло в поздний допустимый срок, а последующие работы хотят выполнить в ранние сроки. Фактически независимый резерв времени имеют лишь те работы, которые не лежат на максимальных путях, проходящих через их начальные и конечные события.

Таким образом, если частный резерв времени первого вида может быть использован на увеличение продолжительности данной и последующих работ без затрат резерва времени предшествующих работ, а свободный резерв времени – на увеличение продолжительности данной и предшествующих работ без нарушения резерва времени предшествующих работ, то независимый резерв времени может быть использован для увеличения продолжительности только данной работы. Работы, лежащие на критическом пути, так же как и критические события, резервов времени не имеют.

Лекция №11-12

Раздел №4. Основы комбинаторики

Содержание: Комбинаторика и комбинаторные объекты. Решение задач с комбинаторными объектами. Размещения, перестановки, сочетания без повторения элементов. Размещения и сочетания с повторениями элементов. Примеры. Треугольник Паскаля. Биноминальные коэффициенты. Производящая функция. Асимптотические оценки комбинаторных объектов. Примеры применения комбинаторики.

Комбинаторика и комбинаторные объекты. Решение задач с комбинаторными объектами

Комбинаторика работает с специальными комбинаторными объектами, которые относят к дискретной математике.

Комбинаторные объекты и комбинаторные числа

В комбинаторном анализе изучаются различные объекты, порождаемые элементами из конечного множества $A=\{a_1,a_2,...a_n\}$, и числовые характеристики этих объектов. Часто рассматриваются, например, упорядоченные или неупорядоченные подмножества множества A, подмножества с повторяющимися элементами из множества A и т. д. Вместе с классами таких комбинаторных объектов естественным образом вводятся и так называемые комбинаторные числа, задающие число объектов в том или ином классе и зависящие от некоторых параметров, например, от мощности исходного множества A и мощности рассматриваемых подмножеств множества A ***(мощность конечного множества равна числу элементов множества)***.

Пусть есть некоторое конечное множество элементов $A=\{a_1, a_2, ..., a_n\}$. Рассмотрим наборы элементов — подмножеств $\{a_{j1}, a_{j2}, ..., a_{jk}\}$ где все элементы принадлежат А для разных наборов j = 1, 2, ..., r. Такие наборы принято называть выборкой объема r из n элементов. Любое подмножество A является выборкой, но не всякая выборка является подмножеством A, так как в выборку один и тот же элемент может входить несколько раз (в отличие от подмножества).

Рассмотрим основные способы формирования выборок.

Определение. **Выборка называется упорядоченной, если в ней задан порядок следования элементов. Если порядок следования элементов несущественен, то выборка называется неупорядоченной.**

Из определения следует, что две упорядоченные выборки, состоящие из одних и тех же элементов, но расположенных в разном порядке, являются различными.

Комбинаторные задачи связаны с подсчетом числа выборок объема r из n элементов, где выборки подчиняются определенным условиям, т.е. выбор производится по какому-нибудь принципу. Подсчет числа выборок основывается на двух правилах теории множеств.

Обозначим **card** ***A*** *— мощность множества A (число элементов).*

Принцип суммы**:** если **card** $\boldsymbol{A = m}$**, card** $\boldsymbol{B = n}$ **и** $\boldsymbol{A \cap B = \varnothing}$, то

card $\boldsymbol{A \cup B = m+n}$. На комбинаторном языке это означает: если объект *A* можно выбрать *m* способами, объект *B* другими n способами и их одновременный выбор невозможен, то выбор “*A* или *B*” может быть осуществлен *m*+*n* способами. Этот принцип дает возможность подсчитать число независимых событий как сумму чисел каждого из событий по отдельности.

Принцип произведения**:** ***если card A=m, card B=n, то card (A×B)=m+n. Здесь A×B означает декартово произведение множеств, что нужно понимать как множество всех пар элементов, в которых первый элемент выбран из A, а второй из B. На комбинаторном языке это означает: если объект A может быть выбран m способами, при любом выборе A объект B может быть выбран n способами, то выбор “A и B” может быть осуществлен m·n способами.***

Размещения, перестановки, сочетания без повторения элементов

Перестановки. Упорядоченные выборки, объемом *n* из *n* элементов, где все элементы различны, называются перестановками из *n* элементов. Число перестановок из *n* элементов обозначается $\boldsymbol{P_n}$.

Теорема. $\boldsymbol{P_n = n!}$

Док-во: Если первый элемент выборки можно выбрать из **n** вариантов, то второй уже только из **n-1** оставшихся вариантов и т. д. Последний элемент берется уже совсем без вариантов. По принципу произведения получаем, что число всех вариантов $\boldsymbol{P = n*(n-1)*(n-2)*...*2*1= n!}$ ч.т.д.

Размещения. Упорядоченные выборки объемом $\boldsymbol{m}$ из $\boldsymbol{n}$ элементов ($\boldsymbol{m} < \boldsymbol{n}$), где все элементы различны, называются размещениями. Число размещений из *n* элементов по *m* обозначается .$\boldsymbol{A_n^m}$

Теорема. $\boldsymbol{A_n^m = n!/(n-m)!}$

Док-во: Первый элемент выборки можно выбрать из **n** вариантов, второй уже только из **n-1** оставшихся вариантов и т. д. Последний элемент берется из уже **n-m** вариантов, так как это как раз **m** вариант по очереди. По принципу произведения получаем, что число всех вариантов $\boldsymbol{P = n*(n-1)*(n-2)*...*(n-m)= n!/(n-m)!}$ ч.т.д.

Сочетания. Неупорядоченные выборки объемом *m* из *n* элементов ($m < n$) называются сочетаниями. Их число обозначается $\boldsymbol{C_n^m}$.

Теорема. $\boldsymbol{C_n^m = n!/[m!*(n-m)!]}$

Док-во: Число сочетаний меньше чем число размещений, так как любое сочетание может быть преобразовано путем перестановок **m** оставшихся элементов. Число перестановок **m!**. По правилу произведения $A_n^m = C_n^m * \mathbf{m!}$ => ч.т.д.

Сочетания, размещения и перестановки являлись подмножествами исходного множества. Рассмотрим выборки, которые не являются подмножествами.

Размещения и сочетания с повторениями элементов. Разбиения

Размещения с повторениями. Упорядоченные выборки объемом m из n элементов, где элементы могут повторяться, называются размещениями с повторениями. Их число обозначается $A_n^m(n)$.

Теорема. $A_n^m(n) = n^m$.

Док-во: Первый элемент может быть выбран n способами, второй элемент также может быть выбран n способами и так далее, m -й элемент также может быть выбран n способами. По принципу произведения получаем n^m .

Перестановки с повторениями. Пусть имеется n элементов, среди которых k_1 элементов первого типа, k_2 элементов второго типа и т.д., k_s элементов s-го типа, причем $k_1 + k_2 + ... + k_s = n$. Упорядоченные выборки из таких n элементов по n называются перестановками с повторениями, их число обозначается $C_n(k_1, k_2, ..., k_s)$. Числа $C_n(k_1, k_2, ..., k_s)$ называются полиномиальными коэффициентами.

Теорема. $C_n(k_1, ..., k_s) = n!/[k_1!*k_2!*...*k_s!]$

Док-во: По аналогии с доказательством теоремы о сочетаниях без повторений можно по правилу умножения представить:
$P_n = C_n(k_1, ..., k_s) *[k_1!*k_2!*...*k_s !]$ так как число перестановок без повторений равно числу перестановок с повторениями умножить на все числа перестановок в каждом варианте перестановок => ч.т.д.

Сочетания с повторениями. Пусть имеется k типов элементов, каждый тип содержит не менее m одинаковых элементов. Неупорядоченная выборка объемом **m** из имеющихся элементов (их число $n \geq m*k$) называется сочетанием с повторением. Число сочетаний с повторениями обозначается $C_n^m(n)$

Теорема. $C_n^m(n) = C_{n+m-1}^m$ ***(без доказательства)***

Примеры применения комбинаторики

***Пример* 1**. A = 10 {различных шоколадок}, B = 5 { различных пачек печенья}. Выбор "A или B" означает, что выбирается что-то одно и способов выбора в этом случае будет 15. Выбор "A и B" означает, что выбирается 1 шоколадка и 1 пачка печенья и различных вариантов для такого выбора будет 50.

***Пример* 2.** Бросают 2 игральные кости. Сколькими способами они могут выпасть так, что на каждой кости выпадет четное число очков либо на каждой кости выпадет нечетное число очков?
Пусть m – число возможностей для выпадения четного числа на одной кости, n – число возможностей для выпадения нечетного числа. Здесь $m = n = 3$. По правилу произведения количество выпадения четных чисел, как и нечетных, равно 9. По правилу суммы количество возможностей для выпадения двух

четных и двух нечетных чисел будет 18.

Пример 3. Сколько существует способов, чтобы расположить на полке 10 различных книг? Ответ: 10!

Можно рассуждать иначе. Выбираем первый элемент, это можно сделать n способами. Затем выбираем второй элемент, это можно сделать $(n - 1)$ способами. По правилу произведения упорядоченный выбор двух элементов можно осуществить $n\times(n - 1)$ способами. Затем выбираем третий элемент, для его выбора останется $n - 2$ возможности, последний элемент можно выбрать единственным способом. Мы вновь приходим к формуле: $n(n - 1)(n - r) \ldots 1$.

Пример 4. Группа из 15 человек выиграла 3 различных книги. Сколькими способами можно распределить эти книги среди группы?

Имеем A_{15}^3 =15!/(15-3)! = 15 ·14 ·13 = 2730.

Пример 5. Группа из 15 человек выиграла 3 одинаковых книги. Сколькими способами можно распределить эти книги?

$$C_n^m = n!/[m!*(n-m)!] = C_{15}^3 = 15!/[3!*(15-3)!] = 15*14*13/(2*3) = 455$$

Пример 6. Кодовый замок состоит из четырех разрядов, в каждом разряде независимо от других могут быть выбраны цифры от 0 до 9. Сколько возможных комбинаций?

Число комбинаций $A_n^m(n) = n^m$. Здесь $n = 10$, $m = 4$ и ответом будет 10^4.

Пример 7. Рассмотрим вектор длины m, каждая координата которого может принимать всего 2 значения: 0 или 1. Сколько будет таких векторов? Это есть выборка размещений с повторениями, объемом m из двух элементов $n=2$. $A_n^m(n) = n^m = 2^m$ Ответ: 2^m

Пример 8. Сколько различных слов можно получить, переставляя буквы в слове “математика”?

Решение. Это перестановки с повторениями. Буква “а” входит 3 раза ($k_1 = 3$), буква “м” – 2 раза ($k_2 = 2$), “т” – 2 раза ($k_3 = 2$), буквы “е”, ”к”, ”и” входят по одному разу, отсюда $k_3 = k_4 = k_5 = 1$. Всего букв n=10.

$C_{10}(3,2,2,1,1,1)$ = 10!/[3!*2!*2!]=151200.

Пример 9. В кондитерской имеется 7 видов пирожных. Покупатель берет 4 пирожных. Сколькими способами он может это сделать ? (Предполагается, что пирожных каждого вида ≥ 4).

Число способов будет определяться формулой сочетаний с повторениями при **n=7, m=4**. $C_n^m(n) = C_{n+m-1}^m = C_{7+4-1}^4 = C_{10}^4 = 10!/[4!*(10-4)!] = (7*8*9*10)/(2*3*4) = 210$

Пример10. Пусть $V = \{a, b, c\}$. Объем выборки $m = 2$. Перечислить перестановки, размещения, сочетания, размещения с повторениями, сочетания с повторениями.

1. Перестановки: $\{abc, bac, bca, acb, cab, cba\}$. P_3=3!=6.
2. Размещения: $\{(ab), (bc), (ac), (ba), (cb), (ca)\}$. A_3^2 =3!/2!=6
3. Сочетания: $\{(ab), (ac), (bc)\}$. $C_3^2 = 3!/[2!*(3-2)!] = 3$
4. Размещения с повторениями: $\{(ab), (bc), (ac), (ba), (cb), (ca), (aa), (bb), (cc)\}$. $A_3^2(3) = 3^2 = 9$
5. Сочетания с повторениями: $\{(ab), (bc), (ca), (aa), (bb), (cc)\}$. $C_3^2(3) = C_{3+2-1}^2 = C_4^2 = 4!/[2!*(4-2)!] = 4*3/2 = 6$

Треугольник Паскаля. Биноминальные коэффициенты

***Замечание**. Числа $C_n^m = n!/[m!*(n-m)!]$ называются биноминальными коэффициентами. Из этой формулы следует, что $C_n^m = C_n^{n-m}$*

Другие свойства биноминальных коэффициентов :

$(a+b)^n = a^n+C_N^1a^{n-1}b+.... +C_N^{n-1}ab^{n-1}+b^n$ **— формула бинома Ньютона.**

$C(0,0)=1,\quad C(1,0)=1,\ C(1,1)=1,\quad C(2,1)=2,\ C(2,2)=1,\ C(3,0)=1,\ C(3,1)=3,\ C(3,2)=3,\ C(3,3)=1,\ C(N,0)=C(N,N)=1$

для квадратного уравнения $(a+b)^2 = a^2+2ab+b^2 = C(1,1)*a^2+C(2,1)*ab+C(2,2)*b^2$

Общее свойство $C_N^M=C_{N-1}^{M-1}+C_{N-1}^M$

Доказательство: $C_n^m = n!/[m!*(n-m)!]$, $C_{n-1}^{m-1} = (n-1)!/[(m-1)!*(n-m)!]$,

$C_{n-1}^m = (n-1)!/[m!*(n-m-1)!] \Rightarrow C_{n-1}^{m-1} + C_{n-1}^m =$

$= m*(n-1)!/[m!*(n-m)!]+ (n-m)*(n-1)!/[m!*(n-m)!] =$

$= (m+n-m)*(n-1)!/[m!*(n-m)!] = n!/[m!*(n-m)!]$ ***ч.т.д.***

О. Треугольник Паскаля — бесконечная таблица биномиальных коэффициентов, имеющая треугольную форму, основанная общем свойстве биноминальных коэффициентов.

Треугольник Паскаля для биномиальных коэффициентов

					1						**N**
				1		**1**					**1**
			1		**2**		**1**				**2**
		1		**3**		**3**		**1**			**3**
	1		**4**		**6**		**4**		**1**		**4**
1		**5**		**10**		**10**		**5**		**1**	**5**

Или аналогично в виде коэффициентов C(n,m) где ниже расположенные элементы равны сумме 2-х вышележащих

					C(0,0)						**N**
				C(1,0)		*C(1,1)*					**1**
			C(2,0)		*C(2,1)*		*C(2,2)*				**2**
		C(3,0)		*C(3,1)*		*C(3,2)*		*C(3,3)*			**3**
	C(4,0)		*C(4,1)*		*C(4,2)*		*C(4,3)*		*C(4,4)*		**4**
C(5,0)		*C(5,1)*		*C(5,2)*		*C(5,3)*		*C(5,4)*		*C(5,5)*	**5**

Построение треугольника Паскаля основано на свойстве биноминальных коэффициентов $C_N^M=C_{N-1}^{M-1}+C_{N-1}^M$. Действительно, для заполнения каждого следующего ряда мы просто суммируем 2 вышележащих элемента. Крайние же элементы подчиняются другому правилу C(N,0)=C(N,N)=1. Очевидно, что треугольник Паскаля показывает эффективный метод/алгоритм вычисления биноминальных коэффициентов.

Производящая функция

Полиномом называют функцию вида $P_n(x) = \sum_{k=0}^{n} a_k x^k$. Здесь наивысшая степень ***n*** называется порядком полинома. Полином легко вычисляется (например, по схеме вычисления Горнера), дифференцируется и интегрируется.

Каждый полином характеризуется набором его коэффициентов — конечным множеством $A=\{a_1, a_2, ..., a_n\}$**.** Таким образом, функция полинома однозначно связана с комбинаторным множеством А. Ее принято называть производящей функцией. Полиномами можно заменять (аппроксимировать) практически любую функцию, поэтому можно рассматривать взаимосвязь произвольной функции с комбинаторным множеством А и наоборот. Это дает возможность анализировать многие неравенства, функции, зависимости (например доказать формулу аппроксимации Стирлинга для факториала).
В математике **формула Стирлинга** (также **формула Муавра — Стирлинга**) — формула для приближённого вычисления факториала и гамма-функции.

$$\lim_{n\to\infty} \frac{n!}{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n} = 1,$$

что эквивалентно

$$n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n.$$

Асимптотические оценки комбинаторных объектов

Особенно сложную задачу комбинаторики составляет анализ комбинаторных чисел и формул при больших значениях (асимптотика). Это определяется как зависимость или функция таких величин от параметра N. Можно сравнивать разные функции для одной асимптотики. Так как эти величины есть функции, то можно их сравнивать по асимптотическим оценкам при стремлении параметра N->∞. Нахождение точной зависимости для конкретной задачи достаточно сложно. По этой причине обычно ограничиваются **асимптотическими оценками** этой функции, то есть описанием ее примерного поведения при больших значениях параметра N. При этом для асимптотических оценок используют традиционное отношение O (читается "O большое") между двумя функциями f(n)=O(n). Асимптотические оценки функции при N->∞ можно представить выражением O(f(N)), которое означает const* f(N). Пусть заданы 2 оценки O(f(N)) и O(g(N)). Будем считать, что:

Функция f(N) растет медленнее g(N), если $\lim_{N\to\infty}$ [O(f(N)) / O(g(N))] = 0.

Функция f(N) растет быстрее g(N), если $\lim_{N\to\infty}$ [O(g(N)) / O(f(N))] = 0.

Функции f(N) и g(N) имеют одинаковую скорость роста, если $\lim_{N\to\infty}$ [O(f(N)) / O(g(N))] = const≠0.

Аналогично по асимптотике роста можно говорить о скорости роста функции. Соответственно по функции асимптотики можно различать линейные O(N), квадратичные $O(N^2)$, кубические $O(N^3)$ экспоненциальные $O(e^N)$ и др. асимптотические оценки. Таким образом формула Стирлинга является примером подбора 2 функций с одинаковой скоростью роста. Она оценивает комбинаторный объект с помощью непрерывной функции. Аналогично и для других комбинаторных чисел и формул можно построить непрерывные функции той же скорости роста. С одной стороны это дает метод приближенного вычисления либо функции (если по другому ее вычислить не возможно), либо комбинаторного числа (если функцию вычислить проще). С другой стороны, оценки могут быть использованы для доказательства различного рода неравенств и оценок.